VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Kontextová analýza textu

Context-based Text Analysis

2019 Bc. Lukáš Kubiš

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Lukáš Kubiš**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Kontextová analýza textu
Context-based Text Analysis

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je zmapovat technologie používané pro automatickou analýzu textu z pohledu detekce významu slov v rámci jejich kontextu. Cílem práce je tyto technologie a knihovny propojit pro kontextovou analýzu předložených textů.

Práce bude obsahovat:
1. State of the art.
2. Podrobný popis zvolených knihoven, technologií a algoritmů.
3. Experimenty a jejich vyhodnocení (možno použít tabulky a grafy).
4. Závěr - zhodnocení výsledků.

Seznam doporučené odborné literatury:

[1[Aggarwal, Charu C., and ChengXiang Zhai, eds. Mining text data. Springer Science & Business Media, 2012.
[2] Berry, Michael W. "Survey of text mining." Computing Reviews 45.9 (2004): 548.
[3] Cohen, Aaron M., and William R. Hersh. "A survey of current work in biomedical text mining." Briefings in bioinformatics 6.1 (2005): 57-71.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **doc. Ing. Jan Platoš, Ph.D.**

Datum zadání: 01.09.2017
Datum odevzdání: 12.07.2019

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA Fakulta elektrotechniky a informatiky ID16 OSTRAVA

doc. Ing. Jan Platoš, Ph.D.
vedoucí katedry

prof. Ing. Pavel Brandštetter, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 12. července 2019

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostravě 12. července 2019 .....

Na tomto místě bych rád poděkoval doc. Ing. Janovi Platošovi, Ph.D. za cenné rady a ochotu při psaní této diplomové práce.

Abstrakt

Tato diplomová práce se zabývá zmapováním technologií použitelných k automatické analýze textu z oblasti detekování významu slov v rámci jejich kontextu. Získané poznatky byly využity při implementaci technologií a následné experimentální analýze textu v přirozeném jazyce. V rámci této diplomové práce byly popsány a použity nejmodernější dostupné technologie.

Klíčová slova: Kontextová analýza, MS Azure Cognitive Services, IBM Watson™ Natural Language Understanding, Microsoft Concept Graphs

Abstract

This diploma thesis is focused on mapping of usable technologies for automatic text analysis from the area of words-meaning detection within their context. The acquired knowledge was used in technology implementation and subsequent experimental analysis of text in natural language. In this diploma thesis were described and used the most modern technologies available.

Key Words: Context-based analysis, MS Azure Cognitive Services, IBM Watson™ Natural Language Understanding, Microsoft Concept Graphs

Obsah

Seznam použitých zkratek a symbolů

AI	–	Artificial intelligence - Umělá inteligence
API	–	Application programming interface – Aplikační programové rozhraní
BI	–	Business Intelligence
BLC	–	Basic-level Categorization
HW	–	Hardware – fyzická výbava počítače
IBM	–	International Business Machines – technologická společnost
LINQ	–	Language Integrated Query – dotazovací jazyk
LUIS	–	Language Understanding service – služba porozumění přirozeného jazyka
MS	–	Microsoft – technologická společnost
NLU	–	Natural Language Understanding
SQL	–	Structured Query Language – dotazovací jazyk
SPARQL	–	Simple Protocol and RDF Query Language – dotazovací jazyk
URL	–	Uniform Resource Locator – definice umístění zdrojů informací

Seznam obrázků

Seznam výpisů zdrojového kódu

1 Úvod

V posledních letech je čím dál častěji vidět snaha mnoha společností působících na poli informačních technologií o připojení se k fenoménu Big Data. Spolu s masivním nástupem hromadného robotizovaného zpracovávání dat a DataMiningu se začínají objevovat projekty, které se snaží o pochopení informace a kontextu těchto dat. Motivací k tomuto vývoji je mnoho, například rychlejší a hlavně přesnější vyhledávání, kvalitnější služby asistentů využívající systém Speech-to-text a případně další marketingové účely. Jinými slovy, pomocí kontextové analýzy se snažíme přiblížit lidskému chápání a výkladu informace nesené textem v přirozeném jazyce.

Pod pojmem přirozený jazyk si můžeme představit prostředek komunikace, který vytvořili naši předkové a neřízeně se vyvíjel stovky let do dnešní doby. Z toho důvodu existuje mnoho různých větví a mutací jazyků a žádný z nich není postaven na pevných zákonitostech a pravidlech. Každý lidský jazyk obsahuje řadu výjimek a odchylek od předpokládaného tvaru, proto jej není snadné převést do strojové řeči. Na základě tohoto problému vznikla vědecká disciplína zvaná "Výpočetní lingvistika". Ta popisuje několik možných přístupů k analýze textu.

2 Přístupy k analýze textu

2.1 Statistický přístup k analýze textu (obsahová analýza)

V tomto přístupu se v analýze přistupuje k hledání identifikátoru obsahu textu. V textu jsou označena klíčová slova, u kterých se posuzuje jejich vývoj četnosti výskytu a užití v čase. Jde tedy o jakési statistické určení parametrů textu, který analyzujeme a tímto určujeme jeho charakteristiku. Z tohoto pohledu ale nelze určit povahu textu ani jeho tématika. Zde je výčet parametrů, které nad daným textem analyzujeme:

- Obsah synonym v textu – užívá se slovotvorná, pragmatická a sémantická analýza
- Nalezení homonym – slova se stejným zápisem, ale odlišným významem, užívá se taktéž sémantické a pragmatické analýzy
- Hledání klíčových významů – slova, která odpovídají parametru analýzy a nesou význam textu
- Hledání anaforických referencí – tato slova nenesou význam textu přímo, pouze se odvolávají např. na podstatné jméno z předchozí věty, pro tyto účely opět slouží sémantická a komplexní syntaktická analýza
- Hledání stop-words – slova nenesoucí žádný zásadní význam, jsou zde pouze pro účely syntaktické správnosti, lze je vyloučit.
- Určení víceslovních spojení – slovní spojení se specifickým významem (např.Vysoká škola Báňská), určují se na základě morfologické analýzy, případně doplněné o syntaktickou analýzu

Díky těmto parametrům získáme metriku textu, kterou lze porovnávat s metrikami jiných textů. V této kapitole byly informace čerpány zejména ze zdrojů [1] a [2].

2.2 Kontext a kontextová analýza textu

Kontext můžeme chápat jako informaci, která charakterizuje okolí - podmínky, subjekty, situaci a vzájemné vztahy. Využití kontextu můžeme rozdělit na vyšší a nízké úrovně abstrakce. Pro lepší představu příkladem analýzy kontextu na nízké úrovni může být změna nabídky zboží eshopu na základě volby zákazníka, či změna menu v mediálním sytému automobilu na základě vzniklé situace. To vše funguje díky chápání systému a jeho reakci na předchozí zkušenost s uživatelem. U systémů, kde jsou interakce s uživatelem předem definované, je snadné vyvodit co uživatel očekává, případně kam míří jeho snaha. U vyšší úrovně abstrakce je situace o dost komplexnější. Systém se musí rozhodovat na základě mnoha vstupů, které nemusí být přesně definované a mnohdy nemusí být ani úplné. Praktickým příkladem může být distribuce elektrické energie a

rozvodné sítě. V této oblasti kontextová analýza postupně nachází čím dál tím větší uplatnění. Systém musí brát v úvahu aktuální poptávku zákazníků, skutečnou produkci a také zkušenosti vývoje z předchozího období. Díky těmto vstupům a zkušenostem může vytvářet jednotlivé prognostické scénáře s určitou mírou pravděpodobnosti a pomoci tak s efektivním využitím distribuční sítě. S hledáním kontextu textu je to v mnoha ohledech podobné. Kontext textu je významové spojení mezi částmi textu, případně slovy, které jsou samostatné. Je to jakési zasazení slova mezi slovy ostatními, každé slovo, fráze, věta má tedy jasné umístění ve sledu slov dalších. Pokud bychom nějak razantně tento sled porušili, výsledný kontext celého textu by mohl být odlišný. Na rozdíl od obsahové analýzy analýza kontextová hledá v obsahu textu souvislosti spojené s analyzovaným termínem. K definici tématu používá databázi znalostí, která ke každé entitě přidává její klasifikaci a snaží se najít porovnání vztahů mezi dalšími entitami. Tento přístup se vyvíjel spolu s přístupem statistickým, nicméně pro něj v praxi dlouhou dobu nebylo využití. S nástupem nových technologií se však rychle snižuje cena výpočetního výkonu a praktické nasazení rozsáhlé kontextové analýzy na poli technologií je tedy aktuálnější.

2.3 Lingvistický přístup k analýze textu

Ve srovnání s předchozími přístupy se lingvistický přístup[3] orientuje primárně na kvalitu popisu textového úryvku. Jde tedy o úplný model textu s využitím maxima poznatků o textu a snahu o modelaci do přirozeného jazyka. Aktuálně je tento přístup spíše teoretický a slouží pouze k experimentování s modely řeči.

2.4 Analýza sentimentu

Sentiment je jeden z dalších parametrů zpracování přirozeného jazyka, který určuje celkové vnímání textu. Každý člověk prožívá nějaké emoce, zastává určitý postoj či názor. Tyto vlivy, ať už vědomě či nevědomky, vkládáme do textu, který vytváříme. Sentiment textu hodnotí tyto postoje ve vztahu k popisu událostí. Vytváří kladné či záporné zaujetí posluchače a tím bezprostředně přispívá k chápání textu. Analýza sentimentu se zabývá problematikou přirozeného jazyka a má za úkol určení postoje autora textu k problematice. Základem analýzy je zjištění, zda analyzovaný text tento postoj obsahuje. Pokud lze z textu vyvodit sentiment, text nazýváme subjektivním, v opačném případě jej nazýváme objektivním. Sentiment textu pak dále rozdělujeme mezi:

- **Pozitivní** – kladný postoj k danému tématu
- **Negativní** – záporný postoj k tématu
- **Neutrální** – nelze určit, lze považovat za druh složitějšího subjektivního postoje
- **Bipolární** – obsahuje kladný i záporný postoj najednou

Samotný výpočet celkového indexu sentimentu textového vzorku lze provést několika způsoby:[4]

1. Absolutní proporcionální rozdíl:
   $Indexsentimentu = (Pw - Nw)/(Pw + Nw + Ow)$
   Nevýhodou této rovnice může být zohlednění slov neutrální povahy, které v některých případech mohou zapříčinit zkreslení výsledku.

2. Relativní proporcionální rozdíl:
   $Indexsentimentu = (Pw - Nw)/(Pw + Nw)$
   Finální výsledek indexu věty může v některých případech inklinovat k jednomu z extrému bez zohlednění ostatních neutrálních slov.

3. Relativní proporcionální rozdíl:
   $Indexsentimentu = log(Pw + 0,5) - log(Nw + 0,5)$
   Funkce je symetrická v okolí 0 a má hladší průběh.

Kde:
Pw – celkový počet slov označených pozitivním sentimentem
Nw – celkový počet slov označených negativním sentimentem
Ow – celkový počet ostatních slov neutrální povahy

Spolu s určením výsledného sentimentu je potřeba určit s jakou pravděpodobností tento sentiment analyzovaný text obsahuje. Problematikou analýzy sentimentu se zabývá řada algoritmů strojového rozpoznávání (např. BoosTexter, AdaBoost), které pracují na principu porovnávání textu se vzorkem ukázkových dat. Jejich přesnost se odvíjí od velikosti naučených dat obsažených v databázi znalostí algoritmu. V současné době tyto algoritmy pracují zejména s anglickým jazykem, nicméně s rostoucí popularitou rozpoznávání přirozené řeči postupně začínají přibývat i další jazykové sady.

2.5 Využití kontextové analýzy

Díky kontextové analýze jsme schopni určit jedinečnou charakteristiku analyzovaného článku/úryvku. Tato znalost je praktická nejen z pohledu určení konceptu a celkového tématu textu, ale lze ji taktéž využít k porovnávání různých textů mezi sebou. Standardní nekomplexní algoritmy jsou schopny porovnat statistické parametry textu, jako například frekvenci frází, jejich posloupnost, délku textu atp. Problém nastává v případě porovnání textu různých jazykových mutací nebo záměnu slov v textu. V těchto případech je možné použít k porovnání výsledky kontextových analýz obou textů. Podobný postup lze použít i v případě, že chceme analyzovat text, který by mohl být kombinací dvou textů (dvou různých konceptů)[4].

2.6 Řešení zabývající se rozsáhlou analýzou textu

V poslední době se tématu rozboru přirozeného jazyka začali věnovat velcí hráči IT průmyslu i nové projekty (startupy). Převážná většina řešení využívá principy AI (Artificial Intelligence –

Umělá inteligence). Za pomocí procesů AI byly sestaveny modely učení a získávání informací z přirozeného jazyka, které slouží k budování databáze znalostí. Díky tomuto přístupu je možné sestavit rozsáhlou sadu dat zahrnující také vzory chování a vzájemné vazby mezi entitami. K výhodám řešení vybudovaných na AI patří také to, že spolu s možností rozeznávání přirozeného jazyka a digitálním zpracováním obrazu mohou využívat v podstatě úplně všechny dostupné zdroje informací, jako je například mluvené slovo, video, obrazové soubory, bioinformatická data atd. S ohledem na obrovské množství nasbíraných informací, které tyto systémy používají a také z důvodu synchronizované řízené aktualizace dat, jsou vesměs všechna řešení vybudována na Cloudové platformě. Ta umožňuje nejen řízení správy dat, ale také otevírá možnosti sdíleného výkonu a efektivní využití HW a také snazší nasazení nových řešení. Pro účely této diplomové práce byly zmapovány a užity technologie společnosti Microsoft a IBM. V závěru práce je v praktické části uveden příklad implementace popsaných technologií.

3 Projekty zabývající se analýzou textu

3.1 Microsoft Concept Graphs

Microsoft Concept Graphs[9] je jeden z prvních projektů společnosti Microsoft zabývající se strojovým chápáním lidské komunikace. Projekt obsahuje aktuálně kolem 12,5 mil. unikátních instancí a 87,6 mil. relací získaných strojovým učením z různých webových stránek. Díky tomu bylo možné sestavit model a znalostní databázi, která slouží jako zdroj informací pro klasifikaci. Samotné odhadování kontextu je silně závislé na kvalitě znalostí obsažených v databázi, která je užita. Dalším podstatným parametrem výsledné funkce algoritmu je i schopnost klasifikovat jednotlivé části textu. Primárně jde o vlastnost algoritmu, která například v případě posloupnosti čísel a jejich formátu, je schopna určit zda se jedná o datum, telefonní číslo nebo jiný číselný údaj. Dále je potřeba, aby algoritmus byl schopen posoudit pravděpodobnost správné klasifikace části textu v souvislosti s ostatními detekovanými výrazy. Příkladem mohou být mnohoznačná slova nesoucí různé významy v závislosti na kontextu. V ukázce níže je demonstrace posouzení slova „Apple“ podle různých metrik měřících pravděpodobnost správnosti odhadu:

Score by P(c|e)		Score by MI		Score by P(e|c)		Score by NPMI		Score by PMI^K		Score by BLC	
fruit	0.415	fruit	0.557	design oriented firm	0.1	fruit	0.111	fruit	0.164	fruit	0.58
company	0.286	company	0.188	commercially important fruit tree specie	0.1	silicon valley titan	0.107	silicon valley titan	0.162	fresh fruit	0.076

Obrázek 1: “Apple” – v základu odlišný pohled na klasifikaci

Pravděpodobnost správnosti odhadu (score) udává numerický vektor distribuce ve vektorovém prostoru daného konceptu. Pokud si tedy pod každým výrazem, který chceme klasifikovat, představíme nekonečné množství aspektů, na základě kterých výslednou klasifikaci provádíme, můžeme tyto klasifikace zaznačit do struktury grafu s různě velkou délkou hran. Danou klasifikaci potom můžeme dynamicky přiřazovat s tím, jak se postupně celý koncept utváří. Jak je v ukázce viditelné, score jednotlivých klasifikací jsou různorodá a nelze z nich nic vyvozovat. Pokud ale doplníme slovo další, které má již docela jednoznačnou klasifikaci a hledáme tedy již slovní spojení „Apple iphone“, výsledný koncept se začíná orientovat směrem k mobilním technologiím.

Score by P(c|e)		Score by MI		Score by P(e|c)		Score by NPMI		Score by PMI^K		Score by BLC	
device	0.241	smartphone	0.205	cheap complex device	0.1	smartphone	0.11	smartphone	0.161	smartphone	0.294
mobile device	0.216	mobile device	0.204	suitable smartphone	0.1	smart phone	0.106	ios device	0.136	smart phone	0.181

Obrázek 2: “iphone” – klasifikace výrazu

Score by P(c|e)		Score by MI		Score by P(e|c)		Score by NPMI		Score by PMI^K		Score by BLC	
smartphone	0.203	smartphone	0.23	emerging graphical user interface	0.1	original mobile phone	0.108	emerging graphical user interface	0.15	original mobile phone	0.314
mobile device	0.19	mobile device	0.143	incompatible phone	0.1	emerging graphical user interface	0.105	original mobile phone	0.139	emerging graphical user interface	0.153

Obrázek 3: "Apple iphone" – změna koncepce slovního spojení

Mluvíme tedy o nějakém modelu grafu, který se přetváří s každým přibývajícím slovem ve větě a tím patrně nejsprávnějším konceptem by měly být uzly, které vykazují jistou podobnost navzájem v kompletním grafu věty. Microsoft Concept Graphs skýtá výhodu právě v možnosti jednoduše přiřadit klasifikaci výrazu dle různých klasifikačních technik. API má také možnost vrátit seznam dalších možných klasifikací seřazených podle pravděpodovnosti (score), v jaké je daný výraz použit. Nejefektivnější z těchto technik se jeví BLC (Basic-level Categorization), která výsledek uvádí na rozumné úrovni obecnosti a zároveň neudává příliš specifickou klasifikaci. Oba tyto extrémy mohou zbytečně komplikovat získávání výsledku analýzy. Nicméně Microsoft Concept Graphs obsahuje i další konceptualizační techniky jako MI, PMI, PMIk atp. a jsou taktéž dostupné skrze API.

3.1.1 Implementace Microsoft Concept Graphs

Microsoft Concept Graphs lze do vlastní aplikace integrovat pomocí knihoven Microsoft Concept Graph API 1.0.0, které přináší snadnější práci v komunikaci se serverem a umožňují jednodušeji pracovat s přijatými a odeslanými informacemi. Tím, že Microsoft Concept Graphs je vyvíjen hlavně pro účely výzkumu a akademické účely (podle dostupných informací se ani neplánuje komerční využití), je i samotné nastavení komunikace jednodušší. Pro nastavení klienta je potřeba doplnit API klíč, který je dostupný na stránkách projektu. K požadavku je potřeba poté připojit také typ požadované konceptualizace, konstantu vyhlazení v podobě decimálního čísla a maximální počet konceptů (TopK).

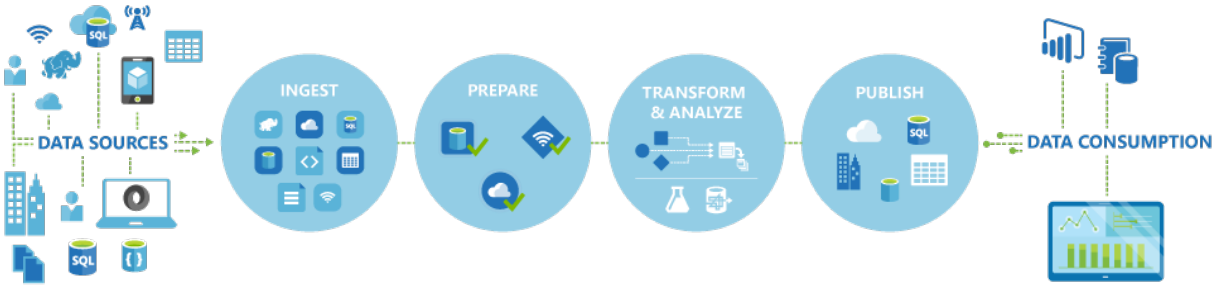
Po zaslání požadavku a jeho zpracování přichází odpověď ve formátu JSON, který lze následně převést například na objekt Dictionary<string, float>. Odpověď obsahuje detekovaný koncept a jeho Score podle požadovaného typu konceptualizace.

3.2 Microsoft Azure

Platforma MS Azure je kompletní cloudové řešení pro běh cloudových aplikací. Jednou z nosných částí MS Windows Azure - Run-time je operační systém sloužící jako prostředí pro spouštění aplikací (aktuálně obsahuje zhruba 600 různých aplikací). Platforma MS Azure ke svému běhu využívá datacentra společnosti Microsoft a poskytuje 3 druhy licenčních modelů (SaaS, PaaS a IaaS). Součástí je prostředí pro vývoj vlastních řešení s podporou poměrně široké škály jazyků (.NET, Python, Java...).

3.3 Microsoft Azure CognitiveServices

Microsoft CognitiveServices je postavena na základu cloudové platformy Microsoft Azure. Obsahuje nástroje AI sloužící k interakci s okolním světem. Příkladem může být rozpoznávání přirozené řeči, jazyka, systém tvorby znalostí, vidění a další podobné systémy. Velká řada z těchto nástrojů je součástí rozsáhlejších AI systémů Microsoftu jako například umělá asistentka Cortana, nebo inteligentní funkce vyhledávače Bing. Platformu Microsoft CognitiveServices užívají i jiné velké společnosti v různých odvětvích průmyslu.



Obrázek 4: Představení prostředí MS Azure CognitiveServices[18]

3.3.1 Rozhraní API pro práci s jazyky

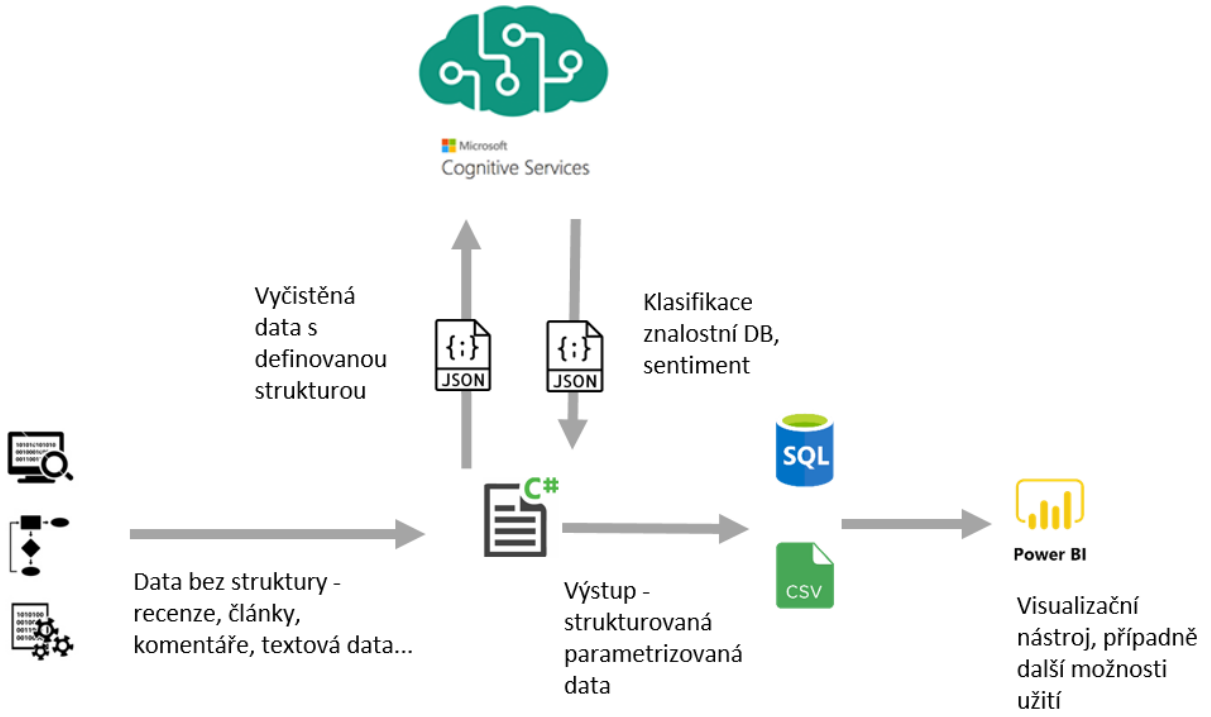
Azure[5] CognitiveServices obsahuje řadu služeb pro sestavení inteligentních aplikací. V této diplomové práci se věnuji rozboru psaného textu přirozeného jazyka, proto se nyní budu orientovat pouze na rozhraní pro práci s jazyky. Celé rozhraní pro práci s jazyky se dá rozdělit na jednotlivé služby, které lze dle potřeb zakomponovat do výsledné aplikace.

Název služby	Definice služby
Kontrola pravopisu (Bing)	Umožňuje provádět kontrolu gramatiky a pravopisu textu
LUIS - Porozumění jazyka	Možnost porozumění lidské řeči (např. chatboti)
Lingvistická analýza	Identifikace struktur textu
Text Analytics	Analýza mínění a zpravcování přirozeného jazyka
Translator text	Překlad textu v reálném čase
Web Language Model	Sekvencování a dokončování slov a řetězců na základě predikce

3.3.2 Analýza textu (Text Analytics)

Pro samotnou analýzu můžeme použít metody, díky kterým lze identifikovat text a určit jeho koncept, případně odhadnout jeho mínění. Metodou Rozpoznání jazyka identifikujeme jazyk, ve kterém je text napsán (aktuálně umožňuje detekci 120 světových jazyků). Výsledkem je sestava s kódovým označením jazyka a také informace ve formě score, která určuje pravděpodobnost určeného jazyka. Následně můžeme provést sestavení a extrakci hlavních bodů v textu. Díky této rychlé detekci z textu na první pohled zjistíme, jaká jsou jeho pravděpodobná témata a jaký je

rozsah klíčových slov. Tato klíčová slova lze dále analyzovat a určit jejich pravděpodobné zařazení do skupin (například, že dané klíčové slovo označuje místo, čas či jinou známou kategorii). Posledním faktorem, který z textu můžeme určit, je jeho mínění. Jinými slovy můžeme určit zabarvení textu – zda je jeho povaha negativní nebo do jaké míry je pozitivní.



Obrázek 5: Diagram principu funkce implementovaného řešení pro sestavení konceptu textu

3.3.3 Implementace MS Cognitive services v .NET prostředí

Implementace služby MS Cognitive services v .NET prostředí spočívá ve správné komunikaci aplikace s API. Příkladem může být případ popisující diagram principu funkce pro sestavení konceptu textu (Obrázek 5). Samotná služba má rozsáhlou podporu programovacích jazyků a je dobře podpořena rozsáhlou dokumentací. Pro navázání komunikace se servery společnosti Microsoft je potřeba ve službě Azure vygenerovat API klíč, který obsahuje referenci na licenci ke službě. Na provedení experimentů jsem využil studentskou licenci pro MS Azure. K zahájení komunikace může posloužit jako příklad třída *ApiKeyServiceClientCredentials* uvedená v příloze.

Následně je potřeba vytvořit klienta pro komunikaci se serverem. K tomuto účelu lze použít knihovnu Microsoft.Azure.CognitiveServices.Language.TextAnalytics, která potřebný interface obsahuje. Pro rychlejší a lepší komunikaci je potřeba u klienta definovat Endpoint, na který se bude připojovat. K dispozici je aktuálně dostatek lokací, na které je možné se připojit. Je zde také možnost měnit lokaci podle dostupnosti nebo výhodnosti připojení. Výčet aktuálních podporovaných lokací je uveden v příloze.

Pro účely experimentu jsem zvolil server westeurope.api.cognitive.microsoft.com a northeurope.api.cognitive.microsoft.com. Nicméně i v případě některých ostatních serverů doba odezvy nebyla zásadně vyšší. Konstrukce samotného klienta za použití třídy *Microsoft.Azure.CognitiveServices.Language.TextAnalytics* je poměrně jednoduchá:

```
// MS Cognitive services - Client
ITextAnalyticsClient client = new TextAnalyticsClient(new
    ApiKeyServiceClientCredentials())
{
   Endpoint = "https://westeurope.api.cognitive.microsoft.com"   //Endpoint
};
```

Výpis 1: Nastavení Endpointu klienta

Pro komunikaci s API je potřeba dodržet strukturu zasílaných dat. Vyžadován je JSON dokument ve formátu – ID, text (případně jazyk textu pro analýzu sentimentu). Velikost je limitována 1000 položkami na jeden JSON dokument s maximálně 5120 znaky. Příklad dokumentu:

```
"documents": [
   {
      "id": "1",
      "text": "This document is in English."
   },
   {
      "id": "2",
      "text": "Este documento está en inglés."
   }
]
```

Výpis 2: Požadavek na server

```
"documents": [
   {
      "id": "1",
      "detectedLanguages": [
      {
      "name": "English",
      "iso6391Name": "en",
      "score": 1
      }]
   },
   {
      "id": "2",
      "detectedLanguages": [
      {
      "name": "Spanish",
      "iso6391Name": "es",
      "score": 1
      }]
   }
]
```

Výpis 3: Odpověď serveru

Samotný typ analýzy volíme funkcí klienta:

```
var result = client.DetectLanguageAsync(new BatchInput(
new List<Input>()
{new Input("3", "This is a document written in English.")})).Result;
```

Výpis 4: Použití analýzy k detekování jazyka

K dispozici jsou analýzy:

1. Detekce jazyka – jako výsledek vrátí název jazyka, jeho kódové značení dle ISO6391 a score, které značí míru určitosti (confidence) daného výsledku:

```
"id": "1",
"detectedLanguages": [
  {
    "name": "English",
    "iso6391Name": "en",
    "score": 1
  }
]
```

Výpis 5: Ukázka vráceného výsledku (JSON)

2. Extrakce klíčových frází – provede detekci a extrakci klíčových slov (keyPhrases) v zaslaném vzorku:

```
 "keyPhrases": [
   "spectacular views",
   "trail",
   "area"
],
"id": "1"
```

Výpis 6: Ukázka vráceného výsledku (JSON)

3. Analýza sentimentu – provede analýzu výsledného sentimentu a vrátí score určující zabarvení vzorku textu v intervalu <0 – 1>. Score v oblasti hodnoty 0,5 značí neutrální zabarvení, hodnoty blíže 0 negativní a krajní hodnota 1 značí pozitivní. V případě, že není možné rozeznat (například vstupem je text v jiném jazyce než je definováno), výsledkem

je neutrální hodnota (0,5)

```
{
   "score": 0.9999237060546875,
   "id": "1"
},
```

Výpis 7: Ukázka vráceného výsledku (JSON)

4. Rozeznání a klasifikace entity – analýza je schopná k jednotlivým retězcům určit jejich klasifikaci na základě znalostní databáze a vrátit k němu parametry jako: délka řetězce, offset, datový typ a subtyp, jazyk, případně referenci na entitu v databázi znalostí Wikipedie formou Id a URL, a také BingId, které lze použít k vyhledání referencí pomocí Bing.com

```
"Entities": [
{
   "Name": "United States",
   "Matches": [
   {
      "Text": "America",
      "Offset": 56,
      "Length": 7
   }
   ],
   "WikipediaLanguage": "en",
   "WikipediaId": "United States",
   "WikipediaUrl": "https://en.wikipedia.org/wiki/United_States",
   "BingId": "5232ed96-85b1-2edb-12c6-63e6c597a1de",
   "Type": "Location"
},
```

Výpis 8: Ukázka vráceného výsledku (JSON)

Pro účely experimentu v diplomové práci jsem využil všechny zmíněné analýzy, nicméně nejzajímavější ze zmíněných je analýza rozpoznání a klasifikace entit textu. Díky této analýze na základě databáze znalostí lze reálně určit obsah a povahu textu. Výhodné mohou být i reference na Wikipedii, případně BingId, kdyby bylo zapotřebí vytvořit referenci na detekované téma.

3.3.4 Power BI platforma

Platforma Microsoft Power BI vznikla jako další produkt Microsoftu určený k práci s daty a jejich analýzou. Tento nástroj je aktuálně licencovaný pro volné použití, tudíž je dostupný všem případným uživatelům. Jeho podstatnou výhodou je rozsáhlá paleta vizuálních nástrojů, které lze použít jako samostatný sdílený dashboard nebo je implementovat do rozsáhlejšího projektu. Příkladem může být prvek WordCloud, kterým lze viditelně vizualizovat výstup z kontextové analýzy. V této diplomové práci jsem MS Power BI využil k vytvoření některých grafických výstupů a také jsem vyzkoušel integraci Microsoft Azure CognitiveServices. V případě jednoduché funkcionality je výhodné aplikaci kompletně vytvořit v prostředí MS Power BI. Problém nastává v případě, pokud chceme integrovat službu, která není podporovaná prostředím (Meaning Cloud, IBM Watson, AmazonComprehend, Google Cloud...) nebo data jsou nekonzistentní. V těchto případech je nutné data předpřipravit a poté je importovat nebo vytvořit napojení na DataSource.

Obrázek 6: Příklad vizualizace výsledku analýzy diskuzního fóra – MS Power BI

3.4 IBM Watson™Natural Language Understanding

Společnost IBM přichází s podobným projektem jako byl popsán u Microsoftu IBM Watson™Natural Language Understanding[6]. Tato řešení mají mnoho podobností, což lze považovat za určitý druh konkurenčního boje. IBM nezůstává pozadu ani s Cloudovými technologiemi a tím pádem Watson™Natural Language Understanding běží jako služba na jejich vlastním cloudu. Z pohledu samotné implementace vlastního projektu lze využít aplikační interface podporované nejběžnějšími jazyky (Java, Python, Go, C#...), případně využít prostředí cloudu k vytvoření služby. Oproti MS Cognitive services má IBM možnost pracovat s více parametry textu. Například kromě sentimentu textu je zde možnost detekce emocí v předloženém obsahu. Rozsáhlejší jsou také metody klasifikace frází a detekce klíčových slov, u kterých lze detekovat vzájemné vz-

tahy. Další možností je využití cloudového nástroje IBM Watson™Knowledge Studio a doplnění standardního modelu o specifické fráze a slovní spojení, které ve standardním modelu nejsou. Tato vlastnost přináší možnost „customizace“ modelu například pro účely použití v prostředí technického odvětví, nebo v případě analýzy textu obsahujícího specifické termíny společnosti. Modely lze libovolně editovat a verzovat.

3.4.1 Implementace IBM Watson™NLU v .NET prostředí

3.4.1.1 Autentizace Pro implementaci služeb IBM Watson™ bylo vytvořeno Watson Developer Cloud .NET Standard SDK. V tomto SDK jsou základní definice pro standardní připojení ke cloudu a komunikaci s API jednotlivých služeb. Jedinou částí, která není zcela standardizovaná, je autentizace. IBM používá 3 typy autentizace, které se liší podle podpory jednotlivých služeb. V praxi to tedy znamená, že se může stát, že pro každou službu budeme potřebovat jinou metodu k vybudování komunikačního kanálu. V příloze jsou blíže zmíněny všechny 3 metody autentizace.

U některých služeb byla doplněna možnost stáhnout po registraci přístupu „Credential file“ (ibm-credentials.env), díky kterému dochází ke sloučení metod přístupu ke službám. Soubor obsahuje JSON strukturu, ze které lze čerpat informace podle identifikované autentifikační metody.

3.4.1.2 API komunikace Pro samotnou komunikaci s API je potřeba mít v projektu vytvořenou referenci na package používané služby:

```
<ItemGroup>
   <PackageReference Include="IBM.WatsonDeveloperCloud.
      NaturalLanguageUnderstanding.v1" Version="3.0.0" />
</ItemGroup>
```

Výpis 9: Přidání reference package

Instalace je možná také přes konzoli Správce rozšíření NUGET:

```
PM > Install-Package IBM.WatsonDeveloperCloud.NaturalLanguageUnderstanding.v1
```

Výpis 10: Přidání reference package pomocí NUGET

Toto rozšíření nám dává možnost užít standardní konstrukci dotazu SDK. K těmto účelům slouží třída NaturalLanguageUnderstandingService, která pomocí http klienta založí komunikaci se serverem. Samotný request obsahuje tag využívané služby (např. "service_name=natural-language-understanding;service_version=v1;operation_id=Analyze"), parametry požadované služby (metadata, relace, entity, sentiment...) a Dictionary customData - objekt obsahující samotná data k analýze.

Na základě požadavku přichází odpověď ze strany serveru ve formě JSON objektu. Objekt po serializaci obsahuje návratové hodnoty k požadovaným službám.

```
if (result.Keywords != null && result.Keywords.Count > 0)
{
foreach (KeywordsResult keywordResult in result.Keywords)
   {
   Console.WriteLine("keywordResult relevance: {0}, text: {1}", keywordResult.
       Relevance, keywordResult.Text);

   if (keywordResult.Emotion != null)
      {
      EmotionScores emotionScores = keywordResult.Emotion;
      Console.WriteLine(string.Format("anger: {0} | disgust: {1} | fear: {2} |
          joy: {3} | sadness: {4}", emotionScores.Anger, emotionScores.Disgust,
          emotionScores.Fear, emotionScores.Joy, emotionScores.Sadness));
      }

   if (keywordResult.Sentiment != null)
      {
      Console.WriteLine("sentiment score: " + keywordResult.Sentiment.Score);
      }
   }
}
```

Výpis 11: Příklad výpisu serializovaného výsledku

Skrze SDK lze komunikovat i v případě použití modelů. Samotné nastavení a editace modelů však možná není.

4 Praktické použití popsaných technologií a experimenty

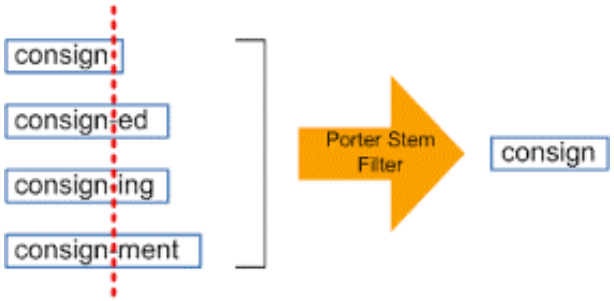
V předešlých částech této diplomové práce popisuji některé z aktuálně dostupných technologií, které mohou být použity k rozsáhlejší analýze textu. Jednotlivé technologie jsem volil podle možností, které aktuálně nabízejí a také podle jejich dostupnosti (licencování a případné užití k edukačním účelům) a podpory v prostředí .NET, popřípadě jiných jazyků. U všech zmíněných technologií jsem se snažil využívat veškeré dostupné funkcionality, nicméně už v průběhu tvorby této diplomové práce docházelo ke změnám některých prvků služeb. Z tohoto vývoje lze usoudit, že obor zpracování přirozené řeči se stále rozvíjí.

4.1 Volba vstupního textu pro experimenty

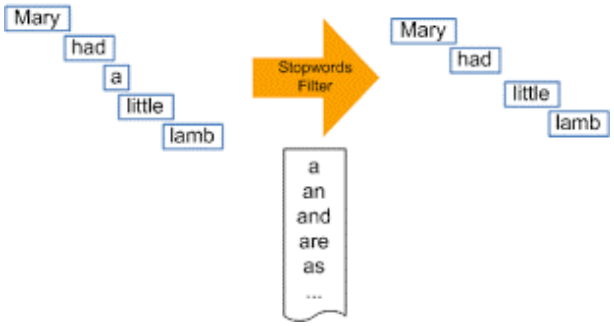
Pro výslednou demonstraci a otestování funkcí analýzy textu bylo nutné použít souvislý vzorek textu, na kterém lze analyzovat koncept a dá se z něj určit jeho zaměření. Z těchto důvodů nelze použít generovaný náhodný text ani sestavu na sobě nezávislých vět. Pro účely analýzy tématu se ukázaly vhodné články popisující jedno téma, popřípadě text vysvětlující nějaký zavedený termín. Díky tomu lze jednoduše posoudit, zda se výsledek analýzy blíží povaze článku. Později, díky možnostem porovnání detekovaných entit se znalostní databází a možnou detekcí známých termínů, jsem se snažil o použití textů například s obsahem známých jmen a názvů společností. Pro větší různorodost bylo výhodné i to, aby text obsahoval označení míst a také nějaké číselné hodnoty včetně formátu data/času. V poslední řadě jsem text záměrně obohatil o URL odkazy, na kterých lze provést analýzu metadat.

4.2 Microsoft Concept Graphs

Vstupem algoritmu je textový soubor, ze kterého je text načten a následně procesován. Před samotnou sémantickou analýzou textu je nutné provést úpravy a převést text do takové formy, abychom mohli provést kontextovou analýzu. Pro zpřehlednění jednotlivých úprav nad textem jsem si zvolil formát tabulky za použití datové struktury DataTable. Souvislý text jsem parsováním rozdělil na menší jednotky – slova. Aby bylo možné zachytit sekvenci jednotlivých slov, každé slovo má pevnou a neměnnou pozici v řádku. Do jednotlivých sloupců tabulky jsem jako atributy umístil jednotlivé fáze úprav nad jednotlivými výrazy a také jejich kontext. První úpravou nad jednotlivými výrazy je lexikální analýza. Jednotlivé výrazy projdou sjednocením velikosti písma, odstraněním znamének bez významu (např. "@", ",", ".", ";", "“"...). Další operací nad textem byl Stemming, při kterém je z jednotlivých výrazů odstraněn prefix a postfix (viz. Obr 7), který není nutný pro pozdější relaci mezi hledaným kontextem výrazu. Pro účely Stemming jsem použil EnglishPorter2Stemmer. Poslední operací nad přípravou textu bylo odstranění tzv. “Stopwords” (viz. Obr 8). Tato slova při použití v textu nenesou žádný význam a jsou vždy charakteristická pro daný jazyk. Pro odstranění jsem vytvořil Regex výraz užívající pole řetězců, do kterého jsem načetl seznam Stopwords pro angličtinu.



Obrázek 7: Ukázka techniky Stemmingu[18]



Obrázek 8: Ukázka odstranění tzv. Stopwords

V této chvíli je již možné plně využít ke klasifikaci jednotlivých klíčových výrazů znalostní databázi MS Concept Graphs. K samotnému připojení ke službě MS Concept Graphs lze využít API. Po navázání komunikace se serverem a nastavení jednotlivých požadovaných parametrů lze získávat klasifikace k zaslaným klíčovým výrazům. Přidání této klasifikace je viditelné na ukázce (viz. Výpis 12), kde po všech úpravách jednotlivých částí věty, je doplněna na konec struktury.

```
and - and - - -
when - when - - -
a - a - - -
coherent - coherent - coherent - coher - proprietary version
written - written - written - write - communicational skill
message - message - message - messag - paraphrase
is - is - - -
```

Výpis 12: Výpis jednotlivých fází úpravy textu a klasifikace dle MS Concept Graph

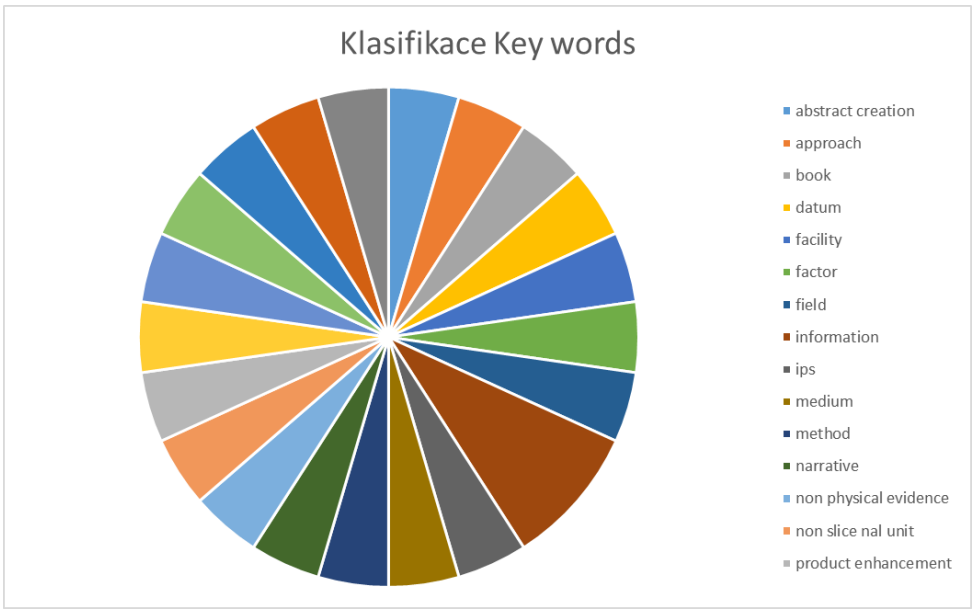
V následující fázi jsem aplikoval dvourozměrnou strukturu naplněnou již pouze jednotlivými klasifikacemi a na základě četnosti jejich výskytu jsem volil pravděpodobný koncept textu. Toto řešení se ukázalo jako použitelné na jednoduché a krátké texty. U dlouhých textů je vysoká rozmanitost v samotné klasifikaci a relativní četnosti (Výpis 13).

```
Frequency of ContentText:                    total key words: 139
     cerebral vocation 3
```

```
cognitive konwledge 2
editing flash element 9
user defined type 2
sense operation1
pentagon planner 3
non standard image type 1
informal urban greenspace 1
traffic control device 2
soft legal document 2
fixed asset 1
adding input 1
yoga prop 1
advanced word processing feature 1
basic need 1
proprietary version 2
structured value 2
midi control 1
bacis scientific observation 1
address event 2
paraphrase 3
language object 1
part of an expression 2
health information function 4
medical treatment 2
art element 1
...
```

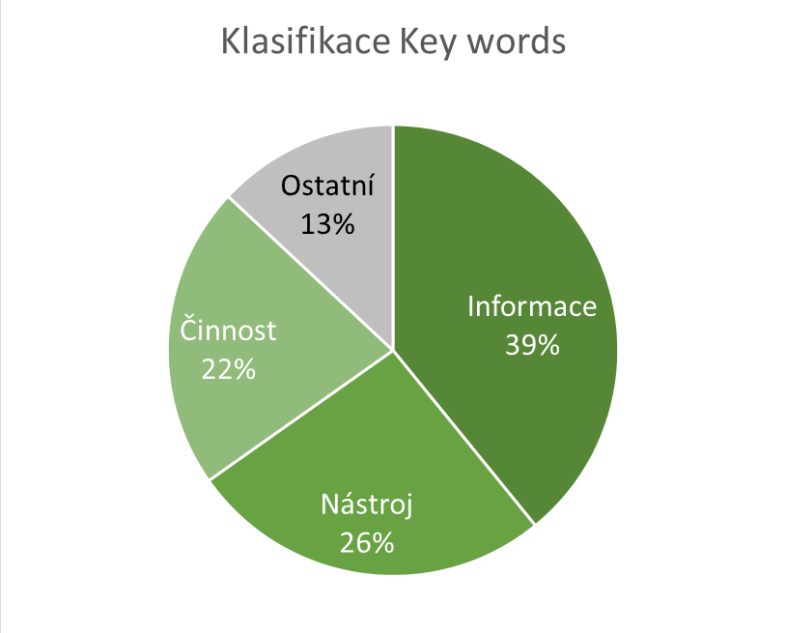
Výpis 13: Výpis výskytů v klasifikaci

Na základě tohoto zjištění jsem svou implementaci doplnil o část kódu, která tvoří dataset pro graf. Dataset je tvořen formou dvojic klíčových slov v ordinálním pořadí. Ke každé této dvojici jsem pomocí Microsoft Concept Graph API doplnil koncept (viz Obrázek 9).



Obrázek 9: Klasifikace Key words

Díky tomuto se zpřesnila analýza u delších textů, nicméně problém velkého množství klasifikací přetrval. Manuálně jsem tedy upravil dataset na základě konceptuálních hodnot a jednotlivé koncepty rozdělil do 3 kategorií (Obrázek 10).



Obrázek 10: Klasifikace Key words do 4 tříd

Díky tomuto rozdělení se dá přehledně prezentovat odhad kontextu článku a orientačně určit, o čem článek pojednává. Experiment byl proveden s článkem pojednávajícím o významu pojmu „Text“ a jeho použití. Dle odhadu kontextu v grafu výsledná analýza toto potvrzuje. Pro efektivnější práci výslednou klasifikací lze použít například ontologické třídy, případně některý z machine learningových algoritmů (např. Naive-Bayes). Microsoft Concept Graphs je jedna z prvních dostupných technologií použitelných ke klasifikaci a analýze konceptu textu. Microsoft Concept Graphs byl vytvořen k experimentálnímu použití, proto aktuálně ale nenabízí tak rozsáhlou paletu možností jako například komerčnější MS Cognitive services a IBM Watson™ NLU.

4.3 MS Cognitive services

MS Cognitive services oproti Microsoft Concept Graphs je schopen pracovat s podporou více jazykových verzí. Nicméně pro účely experimentu jsem volil opět text v anglickém jazyce, což s 99,6% pravděpodobností potvrzuje použitá metoda (TextAnalyticsClient.DetectLanguageAsync) pro detekci jazyka.[7]

```
LANGUAGE IDENTIFICATION
Document ID: 1 , Language: English, en, 0,99612402915
```

Výpis 14: Příklad výstupu detekce použitého jazyka

Z důvodu, že MS Cognitive services má podporu pouze některých jazyků, a také za účelem použití správné jazykové verze znalostní databáze, je nutné tuto informaci o vstupním textu předat serveru. Výsledek detekce jazyka jsem tedy použil jako vstupní parametr pro další analýzy textu. Následně jsem na vstupním vzorku textu provedl extrakci klíčových frází. Vstupní text je potřeba nejprve připravit do formátu podporovaného API. K tomu slouží objekt MultiLanguageInput, který obsahuje kromě samotného textu a parametru jazykové lokalizace také identifikátor požadavku. Výsledkem extrakce je objekt KeyPhraseBatchResult, který lze procházet jako kolekci řetězců.

```
KEY-PHRASE EXTRACTION
Key phrases:  car use
              cars accessible
              Electric cars
              modern car
              American car
              car entertainment
              costs
              land use
              global use
              controls
              century
              driving
              air pollution
              air conditioning
              road congestion
```

Výpis 15: Příklad výstupu po identifikaci klíčových frází nalezených v textu

Samotné klíčové fráze mohou být vhodné například k sestavení lingvistického modelu, případně lze vytvářet nad těmito daty další statistiky. Daleko zajímavější je detekce entit textu, které nám pomohou s klasifikací vzorku. Lze zde v závislosti na typu entity vytvořit referenci na Wikipedii, která detekovaný termín dále rozvíjí. Kolekci entit ukládám do výstupního textového souboru z důvodu lepší manipulace a pro případné další použití. Ke každé entitě jsou v závislosti na jejím typu přidány:

Entity	Wiki - link	Class	SubClass
Car	en.wikipedia.org/wiki/Car	Other	
20th century		DateTime	DateRange
year 1886		DateTime	DateRange
Germany	en.wikipedia.org/wiki/Germany	Location	
Karl Benz	en.wikipedia.org/wiki/Karl_Benz	Person	
Benz Patent-Motorwagen	en.wikipedia.org/wiki/Benz_Patent-Motorwagen	Other	
One		Quantity	Number
first		Quantity	Ordinal
1908		DateTime	DateRange
Ford Model T	en.wikipedia.org/wiki/Ford_Model_T	Other	
United States	en.wikipedia.org/wiki/United_States	Location	
Ford Motor Company	en.wikipedia.org/wiki/Ford_Motor_Company	Organization	
Western Europe	en.wikipedia.org/wiki/Western_Europe	Location	
the decades		DateTime	DateRange
the 2010s		DateTime	DateRange
Fuel	en.wikipedia.org/wiki/Fuel	Other	
Electric car	en.wikipedia.org/wiki/Electric_car	Other	
The New York Times	en.wikipedia.org/wiki/The_New_York_Times	Organization	
Traffic collision	en.wikipedia.org/wiki/Traffic_collision	Other	

Ukázka výsledku klasifikace a vytvoření referencí na zdroje Wikipedie

Jak je vidět na výsledku analýzy entit, text záměrně obsahoval řadu různých datových typů a byl obsahově orientován na pojem „Car“ (automobil) a popisuje počátek automobilové éry. Pro doplnění celkové analýzy jsem provedl analýzu sentimentu. Nicméně vzorek textu je napsán čistě jako popisný a nelze z něj určit pozitivní či negativní zabarvení – výsledek analýzy je tedy neutrální (50%) .

4.4 IBM Watson™ NLU

Obdobným způsobem jako u MS Cognitive services jsem chtěl k analýze použít veškeré možnosti dané služby, nicméně jak bylo již popsáno v obecné části, IBM Watson™ NLU[8] jich nesporně nabízí velké množství oproti již zmíněným řešením. Pro účely experimentu jsem k vytvoření komunikace využil Token a IamApiKey registrovaný přes cloudovou aplikaci IBM Watson. Kromě standardního navázání komunikace se serverem je potřeba sestavit konfiguraci parametrů analýzy pomocí třídy Parameters. Ta obsahuje objekt třídy Features(), ve kterém je možné nakonfigurovat potřebné služby. První analýzu nad textem, kterou jsem implementoval, měla

opět za cíl zjistit jeho obecné parametry jako je lokalizace. Ta stejně jako u MS Cognitive services identifikuje totožný text jako anglický jazyk (vrací označení podle ISO 639-1).

```
LANGUAGE IDENTIFICATION
Language:    en
```

Výpis 16: Příklad výstupu po identifikaci jazyka

Následuje metoda pro identifikaci entit obsažených v textu. Jak je viditelné na ukázce níže, u IBM Watson™ NLU lze docela dopodrobna pracovat s detailními parametry entity. Jako referenční odkaz zde pro změnu používám DbPediaResource. Z hlediska klasifikace, která je možná u jednotlivých entit, je zde možnost až 4 subklasifikací. Tyto klasifikace vycházejí z defaultního modelu znalostní databáze IBM Watson, nicméně je zde možnost přepsání vlastní definicí.

```
entityResult type: Company | relevance: 0.607019 | count: 1 | text: Ford Motor Company
  Result name: Ford Motor Company | dbpediaResource: http://dbpedia.org/resource/Ford_Motor_Company
  subtype: AutomobileCompany
  subtype: AwardWinner
  subtype: Inventor
  subtype: SchoolFounder
--------------------------------------------------------------------------------
entityResult type: Location | relevance: 0.533542 | count: 1 | text: US
  Result name: United States | dbpediaResource: http://dbpedia.org/resource/United_States
  subtype: Region
  subtype: AdministrativeDivision
  subtype: GovernmentalJurisdiction
  subtype: FilmEditor
  subtype: Country
```

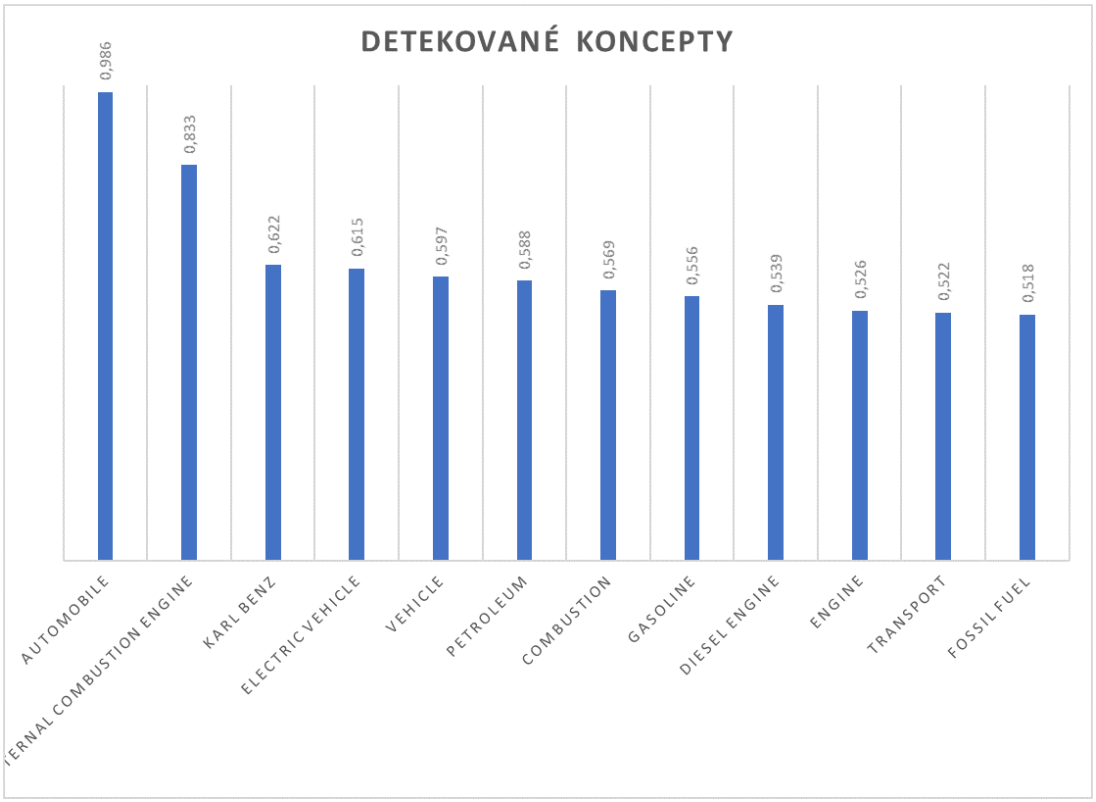
Výpis 17: Příklad výstupu po identifikaci entit textu

Kromě klasifikace jednotlivých entit, lze na základě analýzy sestavit klasifikaci celkového textu. Ta vychází z definované hierarchické struktury klasifikací. Na testovaném textu vychází s pravěpodobností 99,92% klasifikace tématu jako „automotive and vehicles/cars“, což se skutečně shoduje s povahou článku. K výsledku celkové analýzy lze připojit i detekované koncepty. K tomuto účelu používám objekt třídy *ConceptsResult()*, který obsahuje kolekci jednotlivých konceptů spolu s relevancí a referencí na DBpedii.

```
CONCEPT
  Automobile                Relevance 0.9863 DbPediaResource http://dbpedia.org/resource/Automobile
  Internal combustion engine Relevance 0.8329 DbPediaResource http://.../Iternal_combustion_engine
  Karl Benz                 Relevance 0.6225 DbPediaResource http://dbpedia.org/resource/Karl_Benz
```

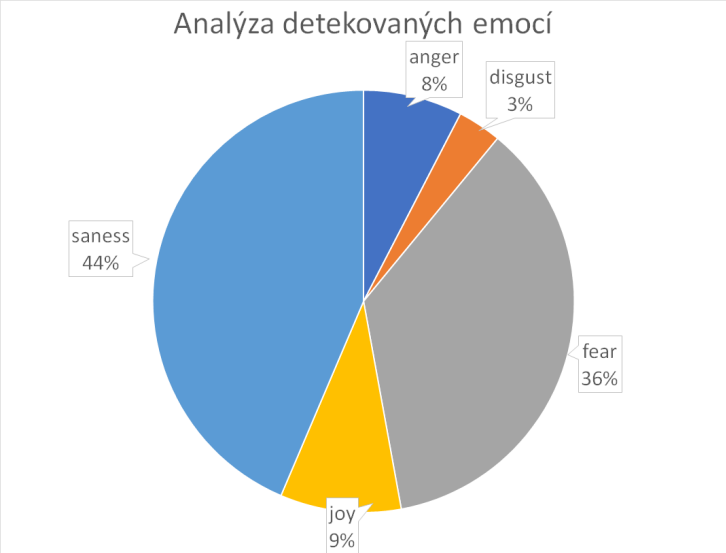
Výpis 18: Příklad výstupu detekce jednotlivých konceptů textu a jejich relevancí

Jak je vidět, zde opět největší význam zaujímá koncept Automobile. Pro úplnost připojuji graf zobrazující detekované koncepty a jejich význam.



Obrázek 11: Srovnání významů jednotlivých detekovaných konceptů v textu

Jak je viditelné v grafu, mezi prvními třemi položkami je strmý průběh a relativně propastný rozdíl. Z toho lze soudit, že koncept „Automobile“ má skutečně nesrovnatelný význam v analyzovaném úryvku. Na závěr přikládám pro zajímavost výsledek analýzy provedené za účelem detekce emocí obsažených v textu. Tento typ analýzy v době vzniku této diplomové práce fungoval v režimu Preview a zmiňuji jej jako jeden z dalších možných rozšíření služeb pro detekci přirozeného jazyka.



Obrázek 12: Srovnání emocí detekovaných v textu

5 Správnost detekce použitého jazyka

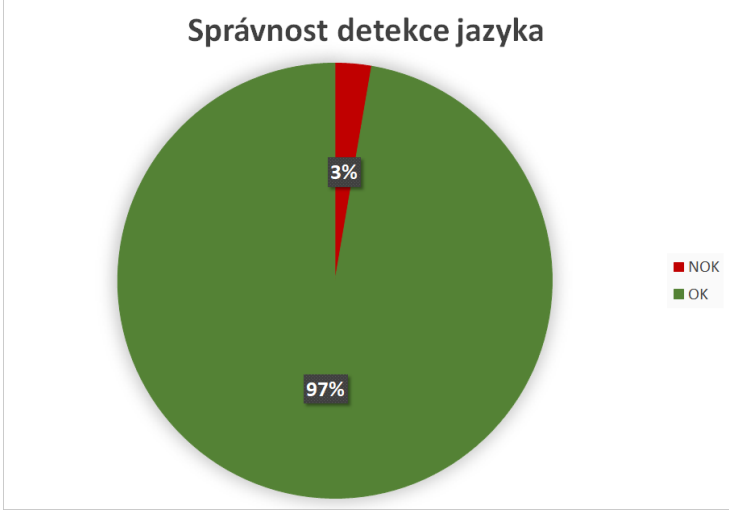
Při implementaci technologie MS Cognitive services jsem jako jeden z prvních parametrů textu určoval použitý jazyk. Z důvodu plné podpory jednotlivých služeb pouze pro anglický jazyk, může být výsledná kvalita analýzy závislá i na tom, zda je celý text napsán pouze v angličtině, či obsahuje další slova, případně celé části napsané v jiném jazyce.

Z těchto důvodů jsem se pokusil ověřit správnost odhadu a pravděpodobnost s jakou je jazyk detekován ověřit. K ověření jsem použil dataset sestavený z krátkých textů v různých jazycích a definici použitého jazyka. Tuto definici jsem později použil jako referenční hodnotu při porovnání výsledku. Dataset jsem pomocí LINQ query načetl do vytvořeného Dictionary ve formátu: *id_vzorku*, *řetězec obsahující vzorek textu*. Pro účely testu jsem zvolil vzorek o rozsahu 1000 řádků. Pro detekci jsem použil službu *TextAnalyticsClient.DetectLanguageAsync()*, u které jsem pro vytvoření pole vstupů použil objekt třídy *TextAnalytics.Models.Input*. Výstup z analýzy jsem exportoval do .csv souboru pro další použití.

```
ITextAnalyticsClient client = new TextAnalyticsClient(new
    ApiKeyServiceClientCredentials())
{
   Endpoint = endpoint
};
var result = client.DetectLanguageAsync(new BatchInput(input)).Result;
```

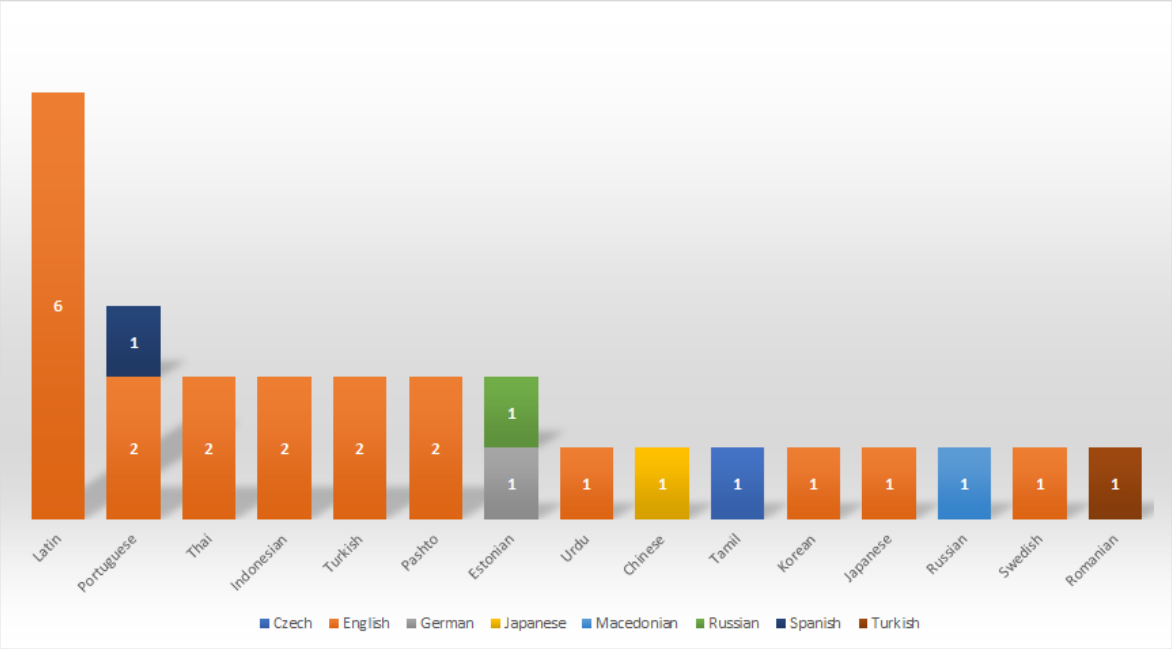
Výpis 19: Vytvoření klienta pro komunikaci se službou pro detekci jazyka

Pro lepší interpretaci jsem výsledek porovnání správnosti detekce jazyka vizualizoval ve formě koláčového grafu. Z grafu je zřejmé, že byla identifikace v 97% správná, zbylé 3% byly složeny z chybně posouzených vzorků.



Obrázek 13: Výsledek porovnání správnosti detekce jazyka textu

Tyto 3% chybných výsledků tvoří celkem 27 chybně detekovaných vzorků. Při pohledu na detailní rozpad chybně detekovaných vzorku můžeme usoudit, že největší procento chyby je přisouzeno latině (6) a portugalštině (3). Zbylých 18 chybných detekcí se dělí mezi 13 různých jazyků. V grafu je na základě podbarvení části sloupců patrné jaký jazyk byl detekován namísto správné varianty. Zde ve 20 případech byl chybně detekován anglický jazyk. Tento případ ukazuje, že v některých případech může mít na kvalitu výsledné analýzy negativní vliv i špatná detekce jazyka.



Obrázek 14: Rozpad chybných výsledků detekce jazyka textu

6 Analýza konceptu a hledání referencí

Na předchozích případech jsem provedl analýzu souvislého článku, na kterém jsem vyzkoušel všechny aktuálně dostupné možnosti analýzy textu v přirozeném jazyce. Na vzorovém textu všechny aplikované analýzy odpovídaly svým výsledkem smyslu textu, program detekoval entity správně a vytvářel reference. Právě u těchto nalezených referencí (tzv. Wikipedifikaci) jsem chtěl posoudit s jakou přesností je tato reference nalezena. Pro účely tohoto testu bylo zapotřebí vytvořit soubor vzorků textu a k nim přiřadit popisované téma s odkazem např. na DBPedii. Jako jeden z dostupných zdrojů pro vytvoření tohoto datasetu se ukázala právě DBPedia, která nabízí k vybranému tématu řadu parametrů. Dataset jsem sestavil ve formátu: *ID* - identifikátor řádku datasetu; *dbo:abstract* - text popisující význam daného tématu, *rdfs:label* - jednoslovné označení tématu. DBPedia pro účely exportování dat nabízí dotazovací rozhraní Virtuoso SPARQL. Zde je možné pomocí dotazovacího jazyku vytvořit jakýkoliv dataset z dat na DBPedii. SPARQL v některých ohledech hodně podobný klasickému SQL, na rozdíl od něj však pracuje nad daty uchovanými ve formátu RDF. Pro účely vytvoření datasetu jsem tedy vytvořil jednoduchou query, která získá z DBPedie 100 jakýchkoliv textů v anglickém jazyce spolu s jejich labelem.

```
PREFIX dbpedia-owl: <http://dbpedia.org/ontology/>

SELECT * WHERE {
?s dbpedia-owl:abstract ?abstract .
?abstract bif:contains "Word" .
FILTER langMatches(lang(?abstract),"en")
}
LIMIT 100
```

Výpis 20: Příklad SPARQL query pro výpis abstraktu a labelu z DBPedie

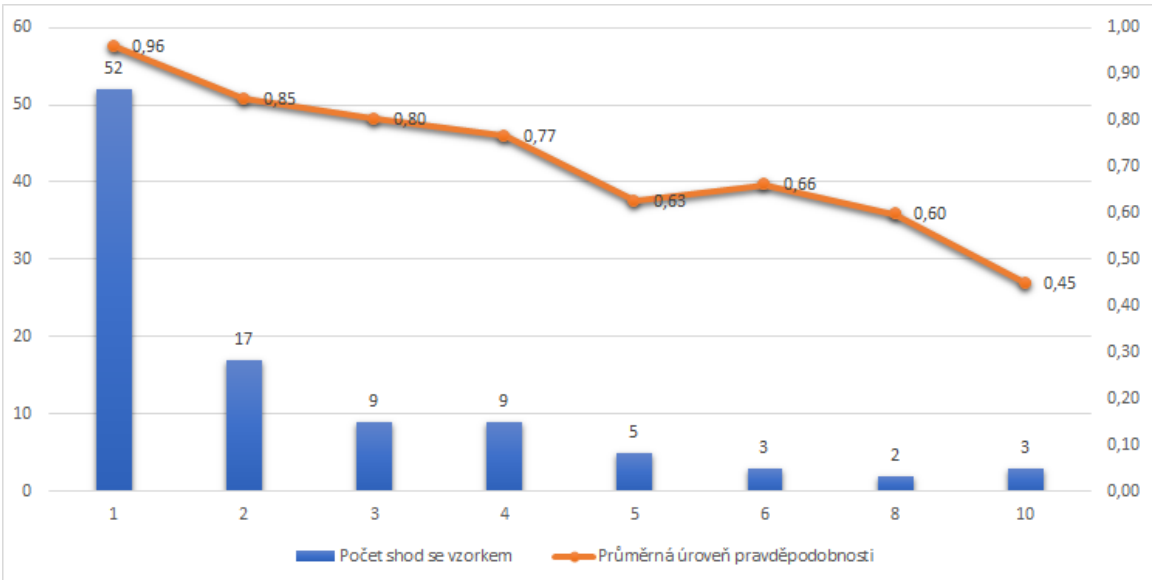
```
var lines = File.ReadLines(pathToCSV);
var textList = lines.Select(line => line.Split(';')).ToDictionary(data =>
    Convert.ToInt32(data[0]), data => data[1]);
```

Výpis 21: Konstrukce LINQ pro načtení potřebných parametrů datasetu

Vytvořený dataset jsem následně načetl pomocí LINQ do Dictionary<int, string> a pro každý text jsem provedl klasifikaci a analýzu detekovaného konceptu textu a jeho referenci na DBPedii. Pro účely této analýzy jsem použil IBM Watson™ NLU services.

Z výstupu analýzy jsem pro každý vzorek textu vytvořil seznam detekovaných konceptů spolu s jejich pravděpodobností. Tyto data jsem následně porovnal s parametrem *rdfs:label* vstupního datasetu. Z detekovaných konceptů a jejich pořadí v seznamu (v závislosti na výsledné

pravděpodobnosti) jsem vytvořil matici, ve které jsem hledal shodu s parametrem *rdfs:label*. Výsledek porovnání uvadím v grafu níže:



Obrázek 15: Graf shody detekovaného koncepu s parametrem *rdfs:label*

Z grafu vychází, že v 52 případech ze 100 je detekce konceptu a následná reference na DBpedii jednoznačná. Následně v 17 případech je plná shoda mezi parametrem *rdfs:label* a konceptem detekovaným na 2. místě. Zbylá část vzorků - 31% se dělí mezi 3. až 10. pozici v seznamu detekovaných konceptů. Dále je v grafu uvedena průměrná pravděpodobnost pro jednotlivé pozice v seznamu konceptů. Pro doplnění grafu jsem se pokusil rozdělit výsledky vzorku mezi úspěšné detekce konceptu (jednoznačná shoda s detekovaným konceptem) a méně úspěšné (2. - 10. pozice v seznamu). U těchto dvou skupin jsem doplnil další faktory, které by mohly mít vliv na určení konceptu. V tabulce níže jsou zmíněny dva faktory, které určují průměrný počet identifikovaných entit ve vzorku a průměrný počet nalezených klíčových frází:

Pozice	**Počet shod se vzorkem**	**Průměrný výskyt entit**	**Průměrný výskyt klíčových frází**
1.	52	3,6	41,6
2. - 10.	48	3,7	35,3

Z tohoto porovnání lze usoudit, že podstatným faktorem při hledání konceptu může být obsah klíčových slov ve vzorku textu a okrajově může mít vliv i výskyt entit, které naopak ve větším množství mohou ovlivnit výsledný odhad směrem ke klasifikaci těchto entit.

7 Analýza výskytu entit v článcích Wikipedie

Podobným způsobem jako jsem ověřil shodu mezi referencí vytvořenou aplikací a datasetem exportovaným dotazem SPARQL z DBpedie, jsem chtěl ověřit správnost nalezených referencí entit pomocí MS Cognitive Services. Jelikož je u MS Cognitive Services pro detekované entity možnost vytvoření reference pouze na článek Wikipedie, musel jsem primárně využít jako zdroj informací právě články Wikipedie.

Narozdíl od DBpedie, kde jsou vazby vytvářeny formou trojic nesoucí další metadata, u Wikipedie jsou vazby mezi články tvořeny odkazem. Tyto odkazy, vložené ve většině případů pravděpodobně manuálně editorem článku, jsou součástí klasického HTML kódu. Z tohoto důvodu bylo nutné pro vytvoření datasetu vytvořit funkci umožňující těžení informací ve správném formátu. Snahou bylo vytvořit dataset obsahující očištěný text článku a k tomuto článku přidružený seznam označených slov s odkazem na jinou stránku Wikipedie. Tyto označená slova považuji za uživatelem vybrané entity odkazující se na jiný článek. Pro parsování článku a získání čistého textu jsem vytvořil metodu *List<string> GetArticleText()*.

```
private static List<string> GetArticleText(string fullUrl)
{
   var htmlWeb = new HtmlWeb();
   var pageHTML = htmlWeb.Load(fullUrl);
   var text = pageHTML.DocumentNode.SelectSingleNode("/html[1]/body[1]/div[3]/
       div[3]");

   var articles = text.SelectNodes("//p");

   var articlesPage = new List<string>();
   foreach (var article in articles)
   {
      if (article.InnerText.ToString()!="")
      {
         articlesPage.Add(article.InnerText.ToString());
      }
   }
   return articlesPage.Distinct().ToList();
}
```

Výpis 22: Metoda pro parsování HTML kódu Wikipedie

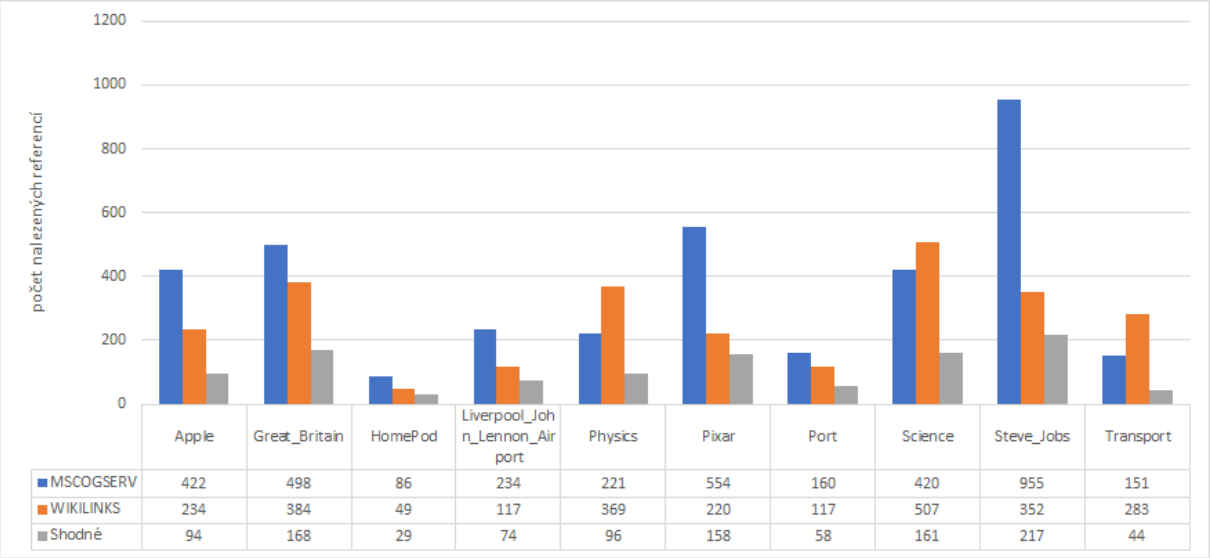
Jak je viditelné z kódu výše, metoda využívá standardní struktury stránek na Wikipedii. Po provedení parsování vrací seznam řetězců obsahující jednotlivé úseky textu. Pro toto dělení

článku jsem se rozhodl z důvodu omezení délky dotazu ze strany API. Abychom měli kompletní informaci k sestavení datasetu, bylo nutné z Wikipedie k získanému textu doplnit i seznam „otagovaných“ slov spolu s referencí na další stránky Wikipedie. Pro tyto účely jsem vytvořil metodu *List<string> GetWikiLinks()*, ta jako výstup vrací seznam nalezených odkazů obsažených v článku Wikipedie.

```
private static List<string> GetWikiLinks(string url)
{
   var webHTML = new HtmlWeb();
   var pageHTML = webHTML.Load(url);

   var text = pageHTML.DocumentNode.SelectSingleNode("/html[1]/body[1]/div[3]/
       div[3]/div[4]");

   var links = text.SelectNodes("//p//a[@href]");

   var linksPage = new List<string>();
   foreach (var link in links)
   {
      var hrefValue = link.GetAttributeValue("href", string.Empty);
      if (hrefValue.StartsWith(@"/wiki/") && !hrefValue.Contains(":"))
      {
         linksPage.Add(hrefValue.Replace(@"/wiki/", "https://en.wikipedia.org/
             wiki/"));
      }
   }
   return linksPage.Distinct().ToList();
}
```

Výpis 23: Metoda pro parsování HTML kódu Wikipedie

Díky těmto dvěma funkcím bylo možné sestavit dataset pro experimentální použití za pomocí technologie MS Cognitive Services. Snahou experimentu bylo detekování entit a vytvoření reference na článek Wikipedie popisující danou entitu ve vzorových textech obsažených v datasetu. Detekované entity jsem následně porovnal s entitami obsaženými v datasetu. Dataset použitý k tomuto porovnání obsahuje kompletní texty z 10 náhodných stránek Wikipedie spolu s celkem 2 632 nalezenými odkazy. Po následné analýze za pomocí MS Cognitive Services bylo celkově detekováno 3 701 entit, ke kterým byly přiřazeny odkazy. V níže uvedeném grafu je viditelné rozdělení počtu detekovaných odkazů mezi jednotlivé stránky Wikipedie:



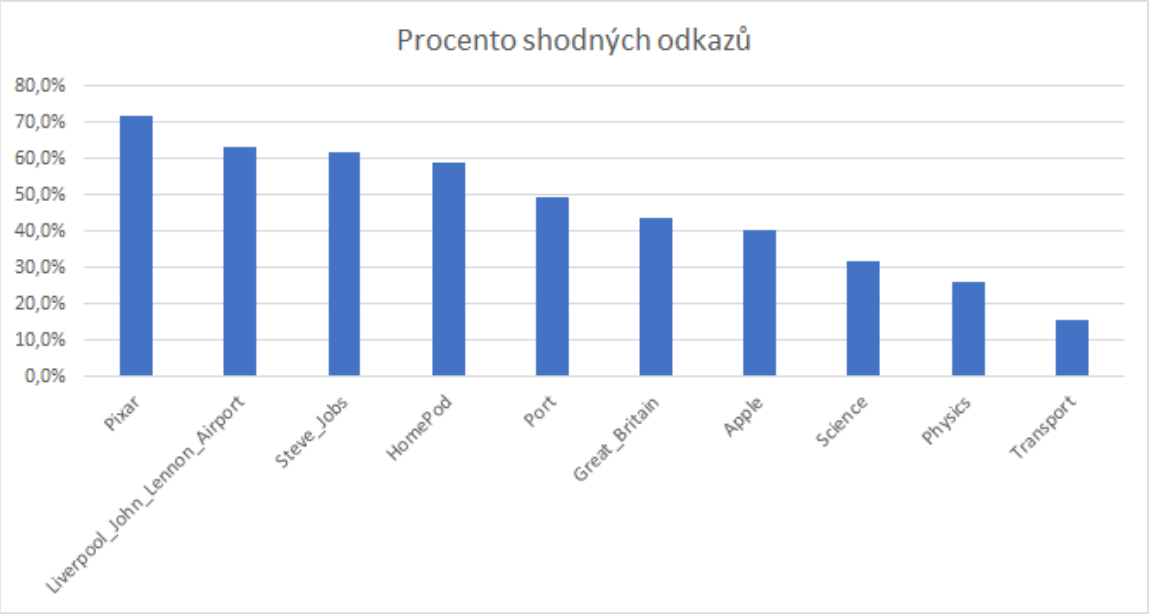
	Apple	Great_Britain	HomePod	Liverpool_John_Lennon_Airport	Physics	Pixar	Port	Science	Steve_Jobs	Transport
MSCOGSERV	422	498	86	234	221	554	160	420	955	151
WIKILINKS	234	384	49	117	369	220	117	507	352	283
Shodné	94	168	29	74	96	158	58	161	217	44

Obrázek 16: Graf počtu nalezených referencí podle analyzovaného článku

V tabulce pod grafem jsou ke každému článku uvedeny počty detekovaných odkazů ve 3 kategoriích:

- **MSCOGSERV** – detekované entity s přiřazením reference za pomocí MS Cognitive Services
- **WIKILINKS** – nalezené odkazy ve zdrojovém článku Wikipedie
- **Shodné** – průnik množin MSCOGSERV a WIKILINKS

Na první pohled je viditelné, že počty odkazů obou množin (MSCOGSERV a WIKILINKS), se liší v závislosti na typu a obsahu článku. Zajímavým zjištěním je průnik obou množin, který ve všech případech dosahuje v průměru pouze na úroveň 41,8 %. To ukazuje, že i když v některých případech MS Cognitive Services detekuje více entit, tyto entity nejsou shodné s těmi, které vytváří autor článku. Jsou zde viditelné rozdíly i v závislosti na článku:



Obrázek 17: Procento shodných odkazů detekovaných MS Cognitive Services

Abych si ověřil, zda byla detekovaná entita a odkaz určen správně, vybral jsem náhodný neshodný vzorek z datasetu a manuálně ověřil jeho správnost. Příklad tohoto ověření je viditelný na obrázku níže. V úseku článku byla detekována entita *3rd millennium BC*, která autorem nebyla odkazována na článek popisující toto téma.

The first forms of road transport involved animals, such as horses (domesticated in the 4th or the 3rd millennium BCE), oxen (from about

3rd millennium BC

From Wikipedia, the free encyclopedia

The **3rd millennium BC** spanned the years 3000 through 2001 BC. This period of time corresponds to the Early to Middle Bronze Age, characterized by the early empires in the Ancient Near East. In Ancient Egypt, the Early Dynastic Period is followed by the Old Kingdom. In Mesopotamia, the Early Dynastic Period is followed by the Akkadian Empire.

Obrázek 18: Příklad neshodného vzorku, u kterého byla vytvořena reference na článek [17]

V opačných případech byl autorem článku označen výraz, který byl sice pomocí Cognitive services detekován jako klíčová fráze, nicméně nebyl určen jako entita. V těchto případech se nabízí možnost MS Cognitive Services využít k detekci klíčových frází a k „otagování“ klíčového slova vyřešit samotnou aplikací. V případě Wikipedie je zde možnost využít struktury URL odkazu jednotlivých stránek a vložit klíčový výraz na konec tohoto odkazu. Pro úplnou funkčnost řešení je vhodné ověřit existenci stránky, na kterou odkaz vytváříme. Příkladem může být jednoduchá konstrukce zmíněná níže:

```
public bool existWeb(string URL)
{
   try {
      WebClient wClient = new WebClient();
      var reqHTML = wClient.DownloadString(URL);
      return true;
   }
   catch (Exception) {
      return false;
   }
}
```

Výpis 24: Metoda pro ověření existence odkazu

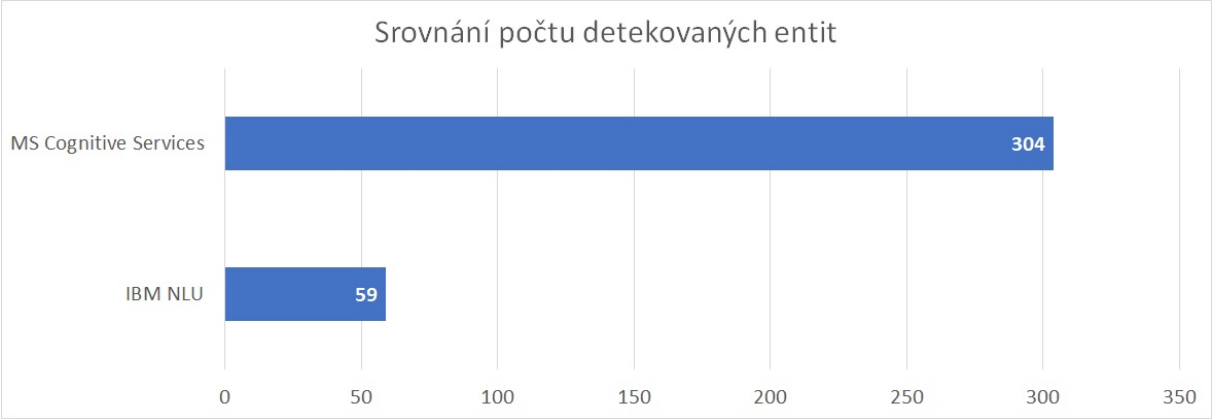
Metoda pomocí vytvořeného požadavku na stažení řetězce z uvedeného URL ověří existenci webu a vrátí bool hodnotu podle výsledku ověření.

Z pohledu implementace „wikipedifikace“ je MS Cognitive Services odlišné od IBM Watson™NLU services. Neliší se pouze zdrojem vrácených referencí (Wikipedia a DBpedia), ale také principem práce s analyzovaným textem. Tyto odlišnosti jsou pravděpodobně způsobeny rozdíly ve skladbě nasbíraných informací. Na základě provedených experimentů lze usoudit, že v případě snahy vytvoření reference na některý ze zdrojů na samotnou detekovanou entitu, je lepší variantou využít MS Cognitive Services. Samotná entita nese ve svém zdroji odkaz na svou definici, což je přesnější a rychlejší varianta, než generování odkazu přímo aplikací. Pokud chceme ale vytvářet reference na detekovaný koncept, je vhodné využít IBM Watson™NLU, MS Cognitive services tuto možnost aktuálně nenabízí.

U detekovaných entit je také možné provést klasifikaci. K nalezené entitě je přiřazena oblast, kterou s největší pravděpodobností popisuje (například označení místa, osoby, organizace, číselného údaje atp.). Experimentálně jsem chtěl ověřit klasifikování entit provedené pomocí MS Cognitive Services. Pro účely porovnání jsem použil službu IBM Watson™NLU, která nabízí obdobu výše zmíněné klasifikace. Pro účely experimentu jsem vytvořil textový vzorek složením několika článků popisujících různé oblasti. Záměrem bylo, aby vzorek obsahoval informace o označení místa, známých textových spojeních jako jsou označení organizací, nebo jména osob. Detailnější specifikace parametrů použitého vzorku textu jsou obsaženy v tabulce níže.

Jazyk	Anglický jazyk
Počet slov v textu	8 116
Počet znaků	42 910
Názvy použitých článků	Apple, Cupertino, California, The United States of America, UNESCO, French, Ford Motor Company

Textový vzorek jsem podrobil analýze entit a jejich následné klasifikace. Výsledkem analýzy byl na první pohled rozdílný počet detekovaných entit v textu.



Obrázek 19: Srovnání počtu detekovaných entit nad vzorkem textu

Ze srovnání je patrné, že MS Cognitive Services má detekci entit mnohem citlivější než srovnávaná služba od IBM. Částečným důvodem tohoto rozdílu je fakt, že každá ze služeb přistupuje k detekci entit rozdílným způsobem. Například IBM Watson™NLU narodíl od Microsoftu neidentifikuje v textu číselné hodnoty a označení data a času. Vstupní vzorek byl z tohoto důvodu pro lepší demonstraci upraven a zbaven těchto hodnot.

Abych byl schopen porovnat samotnou klasifikaci, provedl jsem průnik nad množinou entit detekovaných oběma službami. Množina průniku obou výsledků obsahuje 50 klasifikovaných entit. Pro vizualizaci porovnání jsem vytvořil matici.

Technologie	MS Cognitive Services			
IBM NLU	Klasifikace	[Location]	[Organization]	[Person]
	[Location]	17	3	0
	[Organization]	3	21	1
	[Person]	0	1	6

Sloupce matice pokrývají klasifikaci pomocí Microsoft Cognitive Services, v řádcích jsou uvedeny klasifikaci IBM Watson™NLU. Diagonála matice obsahuje shodně klasifikované entity. Hodnoty mimo diagonálu označují rozdílnou klasifikaci. Po důkladnějším přezkoumání rozdílných hodnot a správnosti klasifikace jsem dospěl k výsledku, že ve 3 případech ze 4 je klasifikace správná podle IBM Watson™NLU. Pouze u jednoho ze zmíněných případů byla klasifikace správná podle MS Cognitive Services.

8 Srovnání použitých technologií a jejich možností

V předešlé kapitole jsem implementoval některé z dostupných technologií použitelných k rozsáhlé analýze textu. V současné době na poli analýzy textu v přirozeném jazyce vzniká řada nových projektů, což potvrzuje lukrativnost tohoto oboru. Nicméně velká většina těchto startujících projektů zaniká, případně je přidružena k projektům zavedených velkých společností. V této diplomové práci jsem se rozhodl použít zejména technologie společností, které na poli IT působí už řadu let. Důvodem byla i následná imlementace, která je snazší u stabilních projektů, než u těch v ranné fázi vývoje.

Pro účely srovnání jednotlivých implementovaných technologií jsem připravil tabulku popisující jejich základní parametry. V tabulce jsou zmíněny metody přístupu, autentizace, jazyková podpora a příklady licencování. Všechna použitá řešení mají k dispozici testovací (Free) licenci s omezením ve formě počtu přenesených informací, Microsoft nabízí i možnost akademického použití.

Služba	MS Concept Graphs	Azure Cognitive services	IBM Watson™NLU
Package - verze	1.0.0	3.0.0	2.17.0
Licencování	Pouze akademické užití - Free	Free do 5 000 transakcí/měsíc, Komerční užití od $0,25 / 1 000 transakcí	Free do 30,000 NLU (NLU - datová jedonotka - 1 NLU = 10 000 znaků), komerční užití bez měsíčního omezení cca $0.003/NLU
Autentizace	API Key	API Key	API Key, Access Token, Account
Administrace přístupů	Ne	Ano (Azure)	Ano (IBM Cloud)
Kategorizace entit	Ano, 6 parametrů (různé úrovně)	Ano, 2 úrovně (Type, Subtype)	Ano, standardně 3 úrovně
Extrakce klíčových frází	Ne	Ano	Ano (limitovaně)
Detekce jazyka	Ne - podpora pouze EN	Ano (až 120 aktuálně podporovaných jazyků)	Ano(package Language Translator.v3 - 62 jazyků)
Detekce sentimentu	Ne	Ano	Ano
Detekce emocí	Ne	Ne	Ano (Preview)
Konceptualizace	Ne	Ne	Ano
Detekce metadat	Ne	Ne	Ano
Detekce sémantických rolí	Ne	Ne	Ano (Preview)
Použití Custom modelů	Ne	Ano (Preview) - pouze	Ano
API podpora	Ne	Ano	Ano
Použití kontejnerů	XML	CSV, JSON	JSON
Jazyková podpora	English	Danish, Dutch, English*, Finnish, French, German, Greek, Italian, Japanese, Korean, Norwegian (Bokmål), Polish, Portuguese, Russian, Spanish, Swedish, Turkish, Danish, Dutch	Arabic, Chinese (Simplified), Dutch, English*, French, German, Italian, Japanese, Korean, Portuguese, Russian, Spanish, Swedish,

*Všechny služby jsou aktuálně dostupné pouze pro anglický jazyk.

U jednotlivých technologií zmíněných v tabulce jsou patrné rozdíly nejen z pohledu licencování, ale zejména z pohledu podpory jazyků a možností, jakým způsobem lze využívat služby spojené se získanými znalostmi. Podstatným aspektem je počet úrovní kategorizace entit, nebo možnost určení jazyka textu. Z pohledu úrovní kategorizace dosahuje nejvyššího počtu tříd MS Concept Graphs. Na základě experimentální analýzy nad vzorovým textem (viz. kapitola 4.2) však přílišné množství tříd kategorizace může zbytečně komplikovat svou detailností výslednou kategorii textu. Tento fakt je způsoben nejspíše snahou o vytvoření databáze obsahující širokou škálu specifických definicí kategorií. Oproti zbylým dvěma technologiím Azure Cognitive services (viz. kapitola 4.3) a IBM Watson™NLU (viz. kapitola 4.4) vyvíjených pro komerční použití slouží MS Concept Graphs spíše akademickým účelům. Na druhou stranu MS Concept Graphs byl jeden z prvních fukčních zdrojů klasifikace a určení konceptu a je možné, že posloužil jako předloha pro vývoj dalších projektů na analýzu textu.

Pro účely určení jazyka vzorku se nabízí díky své podpoře široké škály jazyků jako nejlepší varianta mezi zmíněnými technologiemi Azure Cognitive services (120 podporovaných jazyků). Z těchto důvodů jsem přednostně u této technologie provedl experiment k určení přesnosti odhadu použitého jazyka (viz. kapitola 5). Na základě výsledku testu lze soudit, že s výslednou hodnotou 97 % dosahuje dostatečné spolehlivosti. Technologie MS Concept Graphs bohužel nenabízí dostatečný zdroj informací pro detekci jazyka a tím pádem je orientována pouze na anglický jazyk. IBM Watson™NLU podporuje tuto službu pouze prostřednictvím další služby přidružené ke strojovému překladu, což výrazně komplikuje nasazení této služby do vlastní aplikace. Je totiž potřeba registrace dalšího klíče v IBM Cloud a také je nutné komunikovat přes oddělený aplikační interface - což prodlužuje odezvu aplikace při požadavku analýzy většího objemu různorodého textu.

U IBM Watson™NLU spatřuji největší potenciál v oblasti identifikace entit a hledání konceptu. Tato služba je ve srovnání s dalšími technologiemi pokročilá. Díky možnostem API je možné pracovat v aplikaci s rozsáhlou škálou vrácených parametrů. Pro účely této práce jsem se rozhodl službu identifikace entit a hledání konceptu vyzkoušet (viz. kapitola 6). Výsledné určení konceptu a reference na DBpedii bylo v 52 případech shodné s prvním určeným konceptem, ve zbylých 48 případech byl patrný vliv výskytu klíčových frází a detekovaných entit. Pro účely porovnání jsem provedl experiment na vytvoření odkazu na Wikipedii i za použití MS Cognitive Services (viz. kapitola 7). Výsledek experimentu ukazuje, že pomocí MS Cognitive Services lze obohatit rozsah manuálně vkládáných odkazů autorem článku. Technologie MS Cognitive Services je schopná v průměru vytvořit o 40,6 % více odkazů než bylo vytvořeno samotnými autory. Každá z těchto technologií má odlišný přístup k detekci entit, tudíž zde není možnost plného srovnání. Dá se ale říci, že základní rozdíly jsou patrné už podle využívaných zdrojů odkazu (Wikipedie a DBPedie). Ty samy o sobě mají rozdíly jak ve formě struktury informací, tak principem propojení jednotlivých článků. IBM Watson™NLU navíc oproti MS Cognitive Services nabízí možnost detekce konceptu.

IBM Watson™NLU nabízí také další metody analýzy textu jako například detekce sémantických rolí, sentimentu, emocí atp. Tyto možnosti jsou v současné době dostupné pouze ve verzi Preview a není u nich plně garantovaná funkčnost. Tento vývoj však ukazuje, že tyto projekty jsou stále aktuální a patrně na jejich rozšiřování bude kladen důraz i v budoucnu.

V závěru této kapitoly bych rád zmínil, že každá z technologií má své výhody pro specifický typ implementace. V této kapitole jsem se snažil shrnout možnosti každé z nich a zároveň vyzdvihnout služby s největším potenciálem. Každá z porovnávaných technologií pracuje s jiným rozsahem nasbíraných informací a uživateli nabízí různý rozsah možností v daném sektoru analýzy. Z těchto důvodů je vhodné pro účely implementace v koncové aplikaci zvolit a dobře naspecifikovat potřeby aplikace/projektu a posoudit, která ze zmíněných technologií je nejlepší variantou.

9 Závěr

V této diplomové práci jsem se snažil obsáhnout téma „Kontextová analýza textu“ a popsat možné nástroje, díky kterým ji lze provádět. Práce se z velké části zabývá aktuálními trendy v oblasti práce s daty a zmiňuje cloudová řešení společnosti Microsoft, IBM a s přihlédnutím k principům AI. V práci jsou obsaženy i další rozvíjející se možnosti analýzy textových dat jako například analýza sentimentu, emocí nebo již zavedené metody statistické analýzy textu.

V praktické části projektu jsem implementoval metody analýzy textu v přirozené řeči. Využil jsem při tom nejnovější dostupné technologie společností Microsoft a IBM. Provedl jsem experimentální analýzu na vzorovém článku za použití technologií Microsoft Concept Graphs, Microsoft Azure CognitiveServices a IBM Watson™ NLU. U použitého vzorku textu jsem provedl statistickou analýzu parametrů, klasifikaci na základě znalostní DB u detekovaných klíčových částí s doplněním reference na další zdroje (Wikipedia, DBpedia) a posouzení celkového konceptu článku.

Dále jsem pro účely otestování správnosti analýzy textů uvedenými technologiemi provedl posouzení kvality výsledků. U technologie Azure Cognitive services jsem analyzoval správnost detekce jazyka textu. Na základě datasetu složeného z 1 000 vzorků textů v různých jazycích jsem došel k závěru, že 97% určení je správných. Ve zbylých případech chybných určení na prvním místě figurovala latina mylně detekovaná jako anglický jazyk.

Pro účely druhého testu správnosti určeného konceptu a vytvořené reference na DBpedia.com pomocí IBM Watson™NLU jsem použil jako zdroj vygenerovaný dataset o 100 vzorových záznamech. Dataset byl vytvořen za pomocí dotazu SPARQL z dat DBPedie a byl složený z posuzovaného textu a referenční hodnoty(label textu). Výslednou hodnotu určenou za pomoci IBM Watson™NLU jsem posuzoval s referenční hodnotou datasetu. Výsledkem porovnání jsem došel k závěru, že v 52% byl odkaz na stránku DBPedie a její label totožný s referenčním labelem v datasetu. Ve zbylých případech po důkladnější analýze dalších parametrů byl viditelný vliv poměru výskytu klíčových frází a entit ve vzorcích. Z toho lze usoudit, že při vyšším počtu klíčových frází je algoritmus schopen přesnějšího určení. Naopak výskyt entit ve zvýšené míře může negativně ovlivnit výsledek určení a směřovat jej mylně blíže k samotné entitě.

Poslední experiment pracuje s generovaným datasetem tvořeným parsovaným textem článků Wikipedie a seznamem „otagovaných“ entit obsahující odkaz. Zde se snažím porovnat rozsah detekovaných entit v textu pomocí MS Cognitive Services a entit původních vytvořených autorem článku. Zde jednoznačně ukazuje vyšší průměrný počet detekovaných entit pomocí MS Cognitive Service budoucí možný potenciál těchto služeb na poli automatické tvorby referencí. U detekovaných entit jsem dále experimentálně provedl porovnání správnosti klasifikace entity. Pro účely srovnání jsem využil druhou klasifikaci vytvořenou pomocí IBM Watson™NLU. Výsledky experimentu jsou shrnuty pomocí matice v kapitole **Analýza výskytu entit v článcích Wikipedie**7

Všechny zmíněné experimenty ukazují možnosti a rozdíly jednotlivých technologií. V kapitole **Srovnání použitých technologií a jejich možnosti**8 jsem uvedl srovnání technických parametrů a nekvantitativní porovnání zmíněných technologií na základě implementace.

Pro případné budoucí rozšíření tohoto tématu by bylo přínosné podrobněji analyzovat a doplnit alternativní zdroje získávání informací okolního světa na bázi AI. Konkrétněji se může jednat o metody rozpoznávání znaků, přepis řeči, nebo sestavení jazykového modelu založeného na strojovém učení. Zajímavá může být i implementace popsaných metod spolu s vytvořenými specifickými modely např. ve formě chatbota nebo automatickém čtení životopisů.

Literatura

[1] Svenja Adolphs. Introducing Electronic Text Analysis: A Practical Guide for Language and Literary Studies. Ilustrované vydání. Routledge, 2006. ISBN 9781134361595.

[2] Text Analysis Pipelines: Towards Ad-hoc Large-Scale Text Mining. Ilustrované vydání. Springer. ISBN 9783319257419.

[3] Alexander Gelbukh. Computational Linguistics and Intelligent Text Processing. Ilustrované vydání. Springer, 2003. ISBN 9783540364566.

[4] William Lowe, Kenneth Benoit, Slava Mikhaylov, and Michael Laver. (2011) "Scaling Policy Preferences From Coded Political Texts."Legislative Studies Quarterly 26(1, Feb): 123-155.

[5] Microsoft Azure: Text Analytics [online]. [cit. 2019-04-05]. Dostupné z: `https://docs.microsoft.com/cs-cz/azure/machine-learning/studio-module-reference/text-analytics`

[6] IBM Cloud API Docs: Natural Language Understanding [online]. [cit. 2019-04-15]. Dostupné z: `https://cloud.ibm.com/apidocs/natural-language-understanding`

[7] Data Analysis Expressions (DAX) Reference [online]. [cit. 2019-04-05]. Dostupné z: `https://docs.microsoft.com/cs-cz/dax/data-analysis-expressions-dax-reference`

[8] IBM Knowledge Center: Analyzing Clusters [online]. [cit. 2019-04-05]. Dostupné z: `https://www.ibm.com/support/knowledgecenter/en/SS3RA7_15.0.0/com.ibm.spss.ta.help/tmwb_cluster_intro.htm`

[9] Microsoft Concept Graph: API Usage [online]. [cit. 2019-04-15]. Dostupné z: `https://concept.research.microsoft.com/Home/API`

[10] Zhongyuan Wang and Haixun Wang. Understanding Short Texts, in the Association for Computational Linguistics (ACL). Tutorial. 2016.

[11] Zhongyuan Wang, Haixun Wang, Ji-Rong Wen, and Yanghua Xiao. An Inference Approach to Basic Level of Categorization.ACM International Conference on Information and Knowledge Management (CIKM).ACM –Association for Computing Machinery. 2015.

[12] Zhongyuan Wang, Kejun Zhao, Haixun Wang, Xiaofeng Meng, and Ji-Rong Wen.Query Understanding through Knowledge-Based Conceptualization.IJCAI. 2015.

[13] Wen Hua, Zhongyuan Wang, Haixun Wang, Kai Zheng, and Xiaofang Zhou. Short Text Understanding Through Lexical-Semantic Analysis.International Conference on Data Engineering (ICDE). 2015.

[14] Zhongyuan Wang, Haixun Wang, and Zhirui Hu. Head, Modifier, and Constraint Detection in Short Texts.International Conference on Data Engineering (ICDE). 2014.

[15] Yangqiu Song, Haixun Wang, Zhongyuan Wang, Hongsong Li, and Weizhu Chen. Short Text Conceptualization using a Probabilistic Knowledgebase.IJCAI. 2011.

[16] Wentao Wu, Hongsong Li, Haixun Wang, and Kenny Zhu. A Probabilistic Taxonomy for Text Understanding. ACM International Conference on Management of Data (SIGMOD). 2012.

[17] Wikipedia.org: 3rd millennium BC [online]. [cit. 2019-04-15]. Dostupné z: `https://en.wikipedia.org/wiki/3rd_millennium_BC`

[18] Microsoft Azure: Data Factory - activity overview [online]. [cit. 2019-04-15]. Dostupné z: `https://docs.microsoft.com/en-us/azure/data-factory/copy-activity-overview`

A Konfigurace služeb

A.1 MS Concept Graphs

```
var settings = new ConceptSettings()
{
APIKey = "API-KEY",                              // API Klíč
ScoreFunction = ScoreFunction.ScoreByCross,  // Default ScoreByCross
TopK = 10,                                       // Default 10
SmoothConstant = new decimal(0.00001)            // Default null
};

ConceptClient client = new ConceptClient(settings);

IEnumerable<ConceptInfo> concepts = client.GetConceptsFor("inception"); //
    search concept for some entity
```

Výpis 25: Příklad nastavení klienta

A.2 MS Azure Cognitive Services

```
class ApiKeyServiceClientCredentials : ServiceClientCredentials
{
   public override Task ProcessHttpRequestAsync(HttpRequestMessage request,
       CancellationToken cancellationToken)
   {
      request.Headers.Add("Ocp-Apim-Subscription-Key", "APIKey");
      return base.ProcessHttpRequestAsync(request, cancellationToken);
   }
}
```

Výpis 26: Příklad autentizace u služeb MS Cognitive services

Seznam použitelných koncových bodů:

- West US - westus.api.cognitive.microsoft.com
- West US 2 - westus2.api.cognitive.microsoft.com
- East US - eastus.api.cognitive.microsoft.com
- East US 2 - eastus2.api.cognitive.microsoft.com

- West Central US - westcentralus.api.cognitive.microsoft.com
- South Central US - southcentralus.api.cognitive.microsoft.com
- West Europe - westeurope.api.cognitive.microsoft.com
- North Europe - northeurope.api.cognitive.microsoft.com
- Southeast Asia - southeastasia.api.cognitive.microsoft.com
- East Asia - eastasia.api.cognitive.microsoft.com
- Australia East - australiaeast.api.cognitive.microsoft.com
- Brazil South - brazilsouth.api.cognitive.microsoft.com
- Canada Central - canadacentral.api.cognitive.microsoft.com
- Central India - centralindia.api.cognitive.microsoft.com
- UK South - uksouth.api.cognitive.microsoft.com
- Japan East - japaneast.api.cognitive.microsoft.com
- Central US - centralus.api.cognitive.microsoft.com
- France Central - francecentral.api.cognitive.microsoft.com
- Korea Central - koreacentral.api.cognitive.microsoft.com
- Japan West - japanwest.api.cognitive.microsoft.com
- North Central US - northcentralus.api.cognitive.microsoft.com

A.3 IBM Watson NLU

1. **IAM API Key:**

```
void IAMauth()
{
TokenOptions iamAssistantTokenOptions = new TokenOptions()
{
IamApiKey = "<iam-apikey>",
ServiceUrl = "<service-endpoint>"
};
```

```
assistant = new AssistantService(iamAssistantTokenOptions, "<version-
    date>");
}
```

Výpis 27: Autentizace pomocí IAM API Key

<iam-apikey> - klíč vygenerovaný licenční službou IBM Cloud
<service-endpoint> - definice koncového bodu, volí se podle lokace
<version-date> - verze generovaného klíče

2. **IAM Access Token:**

```
void IAMtoken()
{
TokenOptions iamAssistantTokenOptions = new TokenOptions()
{
IamAccessToken = "<iam-access-token>"
};

assistant = new AssistantService(iamAssistantTokenOptions, "<version-
    date>");
}
```

Výpis 28: Autentizace pomocí Iam Access Token

<iam-access-token> - vygenerovaný neexpirovaný token (většinou jako JSON objekt)
<version-date> - verze generovaného tokenu

3. **Username&Password:**

```
void Passwd()
{
assistant = new AssistantService("<username>", "<password>", "<version-
    date>");
```

```
}
```

Výpis 29: Autentizace pomocí Username&Password

<username>	- username k používané službě (ne k IBM Cloud!)
<password>	- heslo k výše zmíněnému účtu
<version-date>	- verze generovaného účtu